uc3m | Universidad Carlos III de Madrid

Grado en Ingeniería Electrónica Industrial y Automática
2018-2019

Trabajo Fin de Grado

“Diseño y desarrollo de un sistema domótico de bajo costo”

Miguel Vázquez Wilks

Tutor

Ramón Ignacio Barber Castaño

Título: Diseño y desarrollo de un sistema domótico de bajo costo

Autor: Miguel Vázquez Wilks

Tutor: Ramón Ignacio Barber Castaño

EL TRIBUNAL

Presidente:

Vocal:

Secretario:

Realizado el acto de defensa y lectura del Trabajo Fin de Grado el día ..... de ..... de ... en ....., en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de:

VOCAL

SECRETARIO PRESIDENTE

AGRADECIMIENTOS

Quisiera agradecer en primer lugar a mi tutor Ramón Barber sus buenos consejos, su comprensión y su infinita paciencia.

También quiero agradecer a mi familia por estar siempre ahí, por su ayuda y sus ánimos durante toda la carrera; que han hecho posible llegar hasta el final.

A mis compañeros de clase por hacerme disfrutar de esta etapa, y a todos mis profesores por todo lo que me han enseñado durante todos estos años.

RESUMEN

El presente proyecto se basa en el diseño y desarrollo de un sistema domótico de bajo costo orientado a viviendas particulares.

Se desarrollarán módulos para cubrir los distintos entornos de la vivienda: control de luces o enchufes, control de persianas o estores eléctricos, apertura de puertas, seguimiento de temperatura y humedad, control de la calefacción y creación de un último módulo o unidad central para control del sistema completo. En este proyecto se desarrollarán una serie de módulos que permitirán el control de los principales sistemas de una vivienda.

Conforme al espíritu general de este proyecto se pretende que el precio de estos módulos sea lo más asequible posible en comparación con el resto de ofertas existentes actualmente en el mercado.

Palabras clave

Domótica, Control de edificios, inteligente

TABLA DE CONTENIDO

TABLA DE FIGURAS

1. INTRODUCCION

1.1 Motivación del trabajo

En la actualidad se está hablando mucho de la domotización y de los sistemas domóticos en las viviendas, para mejorar la calidad de vida de sus habitantes y mejorar los estándares de habitabilidad de dichas viviendas. En el mercado actualmente existen numerosos sistemas domóticos que permiten el control de todos los elementos básicos de las viviendas, pero normalmente no se comenta que estos sistemas son extremadamente caros y no están al alcance de todo el mundo.

También está el hecho de que la mayoría de los sistemas domóticos interconectados tienen la necesidad de estar comunicados conectados mediante cables, para su control y uso, lo cual requiere de una canalización específica que o se realiza mediante una obra, ya sea en nueva construcción o en vivienda de segunda mano, lo cual encarece el precio de la domotización.

Debido a la necesidad cada vez más imperante de mejorar el ahorro de energía o la eficiencia energética y el aprovechamiento de los recursos naturales que poseemos, se implantará en un futuro muy cercano la necesidad de domotizar las viviendas para un mejor aprovechamiento de estos recursos.

Esta reflexión es la que motiva la realización del presente desarrollo de este proyecto, teniendo en cuenta los tres factores fundamentales, bajo costo, mayor eficiencia energética y facilidad de manejo para el usuario, de forma que se produzca una mejora en la sociedad, el entorno y la persona.

1.2. Objetivos

El objetivo de este proyecto es el de diseñar y desarrollar un sistema domótico funcional de bajo costo el cual no tenga la necesidad de realizar una canalización nueva para la comunicación de los módulos entre sí y de esta manera que sea mucho más atractivo y a la vez asequible para los posibles clientes.

Se ha optado cubrir los siguientes ámbitos:

- Iluminación: Mediante módulos con relés que controlen las luces, y actuadores que controlen la apertura y cierre de las persianas.

- Climatización: Mediante el control del termostato de la vivienda, y la apertura de las persianas.
- Acceso: Mediante el control de la apertura de la puerta de acceso a la vivienda.
- Ahorro energético: Mediante el control de las luces y los enchufes.
- Control: Mediante un módulo central que coordine los controles anteriores.

El sistema permitirá su control desde un dispositivo móvil para que el usuario pueda controlar el sistema desde cualquier lugar de la vivienda.

Asimismo, se permitirá el funcionamiento, de los dispositivos a controlar, de la manera tradicional para que si el usuario no dispone en ese momento de un dispositivo móvil a mano pueda usar todos los elementos.

Aunque uno de los objetivos es el de facilitar y hacer más cómoda la vida al usuario también se busca el producir un ahorro energético en la vivienda mediante el control de los distintos elementos.

Para esto será necesario el desarrollo de un módulo independiente por cada sistema a controlar, estos sistemas pueden abarcar varios ámbitos por lo que no es necesario un módulo específico por ámbito.

1.3 Partes del documento

El presente documento se compone de los capítulos observados en la tabla de contenido:

Ámbito y antecedentes:

- Capítulo 1: Se explica las motivaciones que han impulsado la realización del trabajo, así como los objetivos que se pretenden cumplir a su finalización.
- Capítulo 2: Se pretende dar una visión general de la domótica a nivel de definición, historia y explicar cómo son los sistemas domóticos actuales.
- Capítulo 3: Se desarrollan las regulaciones a las que está sujeto el trabajo, así como la situación socio-económica del ámbito de la domótica.

Diseño y desarrollo:

- Capítulo 4: Se explican los ámbitos de la vivienda que se van a controlar; a su vez se detallan todas las tarjetas que serán utilizadas en el trabajo, así como los periféricos a utilizar, y por último se comentan los aspectos de conexionado de los módulos del sistema a diseñar.
- Capítulo 5: Se expone la parte del software del trabajo y se explica en que consiste el software de cada módulo individualmente.

Resultados:

- Capítulo 6: Se detalla el resultado final del trabajo, así como se exponen los ejemplos de funcionamiento del sistema detallando el funcionamiento de cada módulo.
- Capítulo 7: Se muestra un ejemplo de presupuesto para la instalación de un sistema domótico con los módulos desarrollados en este trabajo, así como el coste real de los elementos utilizados en el proyecto.
- Capítulo 8: Se desarrollan las conclusiones alcanzadas en el trabajo y se exponen algunas de las posibles líneas de trabajo futuras.

Por último, en los anexos podemos observar el código completo que utilizan los módulos del sistema.

2. ESTADO DEL ARTE

2.1. Definición

La Real Academia de la Lengua [1] define la palabra domótica, como "conjunto de sistemas que automatizan las diferentes instalaciones de una vivienda".

Según la Asociación española de Domótica e Inmótica, [2] **"**La domótica es el conjunto de tecnologías aplicadas al control y la automatización inteligente de la vivienda, que permite una gestión eficiente del uso de la energía, que aporta seguridad y confort, además de comunicación entre el usuario y el sistema.**"**

Por lo tanto, la domótica se define: como el conjunto de sistemas inteligentes capaces de automatizar una vivienda, aunque hay muchas más definiciones y se pueden contemplar otras expresiones como podría ser Hogar Digital o Edificios Inteligentes. En el fondo todos estos nombres vienen a decir lo mismo que domótica, ya sea más enfocado a términos de seguridad y conexión o enfocados al control de distintos sistemas de la casa.

2.2 Historia

Se podría considerar que el comienzo de la domótica está en la aparición de los electrodomésticos en el siglo XX ya que son las primeras máquinas que realizan tareas de la casa, sustituyendo las labores realizadas por las personas, aunque a finales del siglo XIX existían ciertas maquinas que realizaban algunas tareas del hogar, pero eran aspiradoras con motores de combustión que se usaban principalmente en la industria.

El inicio de los sistemas domóticos propiamente dichos arranco con los primeros sistemas de climatización, estos sistemas eran sistemas cerrados y con baja velocidad de comunicación, surgiendo el primer protocolo de comunicación enfocado a la domótica fue el X10 [3] el cual salió al mercado en el año 1978.

Seguidamente en los años 80 aparecieron los primeros sistemas propietarios y los primeros buses de comunicación digitales, en los años 90 aparecieron los primeros sistemas abiertos, como LonWorks [4], BacNet [5], Profibus [6].

A principios de los años 90 la popularización de los servicios de tele asistencia conectados a centralitas y su boom con la aparición de internet en 1998, produjeron el arranque real de la domotización en el hogar, produciéndose una transformación paulatina de todos los electrodomésticos adaptándose al internet de las cosas (IoT).

Con la aparición de los smartphones en 2007 y la revolución que produjeron estos dispositivos en internet, el ámbito de la domótica sufre una transformación importante al permitir controlar los servicios domóticos desde cualquier lugar. En la actualidad, gracias a las numerosas apps existentes en el mercado, es posible controlar casi cualquier sistema domótico que tengamos instalado en una casa.

En la actualidad existen numerosos sistemas para la domótica los cuales tienen diversos sistemas de comunicación ya sea mediante protocolos propios o con protocolos libres como: X10 [3], LonWorks [4], KNX [7], Zigbee [8] o EnOcean [9]. Estos sistemas también difieren en los medios de trasmisión, los más comunes son los cableados red, la red eléctrica, las RF, o los cables dedicados y distintas tipologías como pueden ser la centralizada, la de estrella y la de árbol, aunque las más utilizadas actualmente son las de bus y la distribuida.

2.3 Sistemas domóticos

Los sistemas domóticos se componen de tres partes principalmente:

- Los sensores: en los cuales están contenidos todos los módulos que permiten una entrada de información al sistema como son: los pulsadores, los teclados, las pantallas táctiles, los sensores propiamente dichos, las cámaras de seguridad, etc.
- Los actuadores: están compuestos por todos los módulos que permiten las salidas del sistema como pueden ser: las pantallas, las electroválvulas, los relés, etc.
- Los controladores: pueden estar distribuidos y embebidos en cada módulo o estar en módulos especializados en el control del sistema, en este bloque también se incluyen los routers que permiten cambiar de un sistema de comunicaciones a otro, y las pasarelas residenciales que permiten el control de distintas subredes de distintos sistemas de comunicación.

Los sistemas domóticos son muy variados y se enfocan en áreas muy distintas, aunque las áreas en las que tiene más implantación son:

- Iluminación: Control de las luces de la vivienda, ya sea mediante escenas, el control del ciclo día noche o de manera convencional a petición del usuario.
- Consumo eléctrico: Basado en medir y controlar el consumo de energía de la vivienda; como podría ser la desconexión de ciertos sistemas no esenciales si el consumo sobrepasa cierto valor.
- Climatización: Se controla el termostato de la vivienda estableciendo escenas personalizadas, incluido el control de manera independiente de la temperatura de cada habitación.
- Alarmas técnicas: Mediante sensores de: inundación, gas, humo. El control de los distintos sistemas de la vivienda permite avisar de posibles incidentes o peligros

y minimizar el problema cerrando los distintos suministros inmediatamente sin intervención física del usuario.

- Seguridad: El control del acceso a la vivienda y las posibles violaciones del perímetro de la vivienda como pueden ser las roturas de cristales o las aperturas de los accesos de manera indeseada.

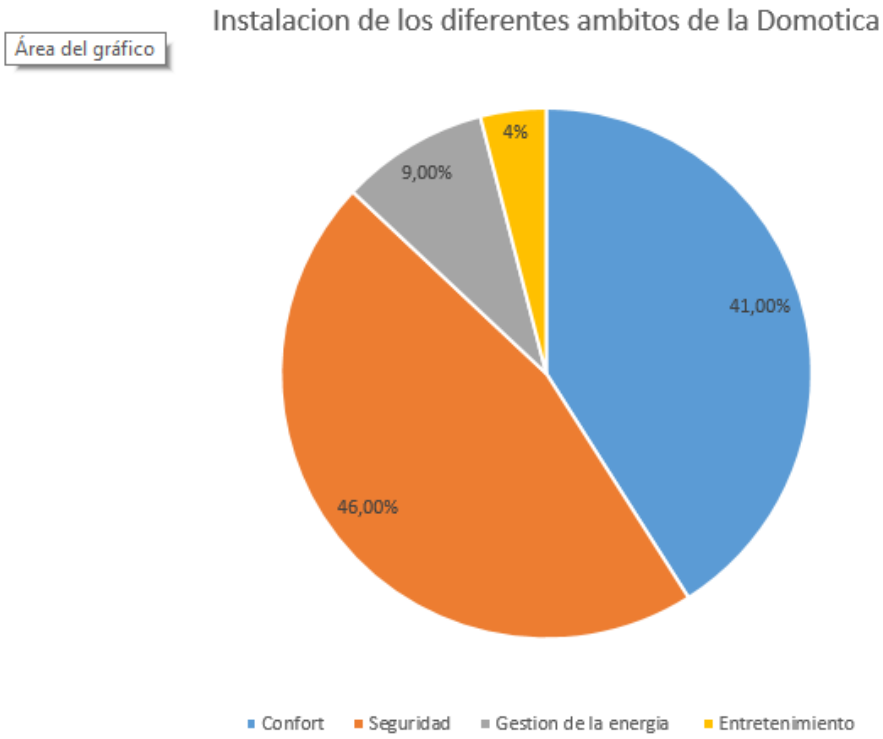

Figura 1. Instalación de los diferentes ámbitos de la domótica

Aunque las áreas expuestas anteriormente son las que reciben un mayor enfoque en sistemas completos existen otras áreas de aplicación que habitualmente se encuentran de manera independiente, como pueden ser:

- Sistemas multimedia: La música y la televisión implementadas mediante altavoces y pantallas repartidas por la vivienda e interconectadas entre sí.
- Sistemas de limpieza autónoma: Como ejemplo los robots aspiradores (roomba).

En los sistemas domóticos se pueden observar distintos medios para la comunicación entre los dispositivos, los principales medios de comunicación de los sistemas son:

- El uso de cable dedicado ya sea un cable específico del sistema o cable ethernet.
- El uso de ondas portadoras o tecnología PLC la cual consiste en enviar la información a través de la red eléctrica.
- El uso de radio frecuencias para la comunicación.

Cada medio tiene sus ventajas y sus desventajas, por lo que en cada caso se deberá optar por el medio de comunicación más adecuado para el usuario.

Existen numerosas tipologías de red, pero en domótica se usan principalmente la de anillo y la de malla. La tipología de malla es la más recomendable para temas de domótica dado que permite el fallo de un dispositivo sin que se rompa la red entera, también permite que los mensajes entre dispositivos busquen la ruta más corta para llegar a su destino, aunque esta tipología no suele utilizarse en los sistemas que se comunican mediante cables dado el costo elevado de conectar cada dispositivo con dos o tres dispositivos a la vez por eso en los sistemas que se comunican mediante cable se utiliza la tipología de anillo normalmente mediante un bus de comunicación.

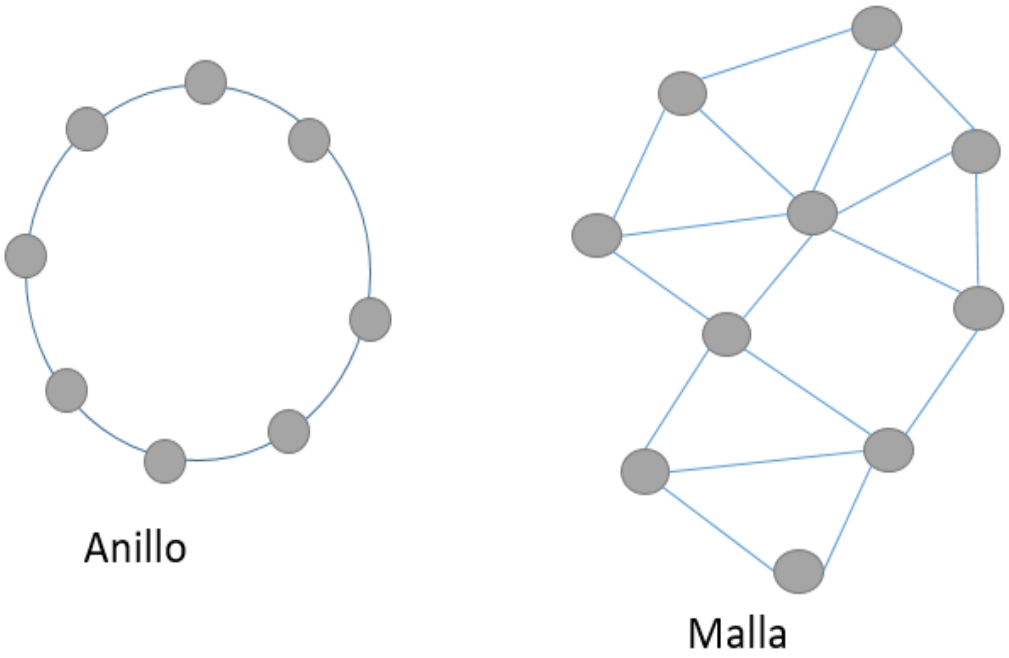

Figura 2. Tipologías de: Anillo y Malla

Si bien es cierto que las dos tipologías comentadas antes son las más comunes, muchas veces no es posible implementarlas, ya sea debido a la imposibilidad de dispositivos de conectarse a más de un dispositivo o por otras razones. En estos casos se suelen utilizar las tipologías de estrella o árbol las cuales ambas son tipologías centralizadas en las que se suele encontrar un dispositivo coordinador que controla todo el sistema y maneja el flujo de información.

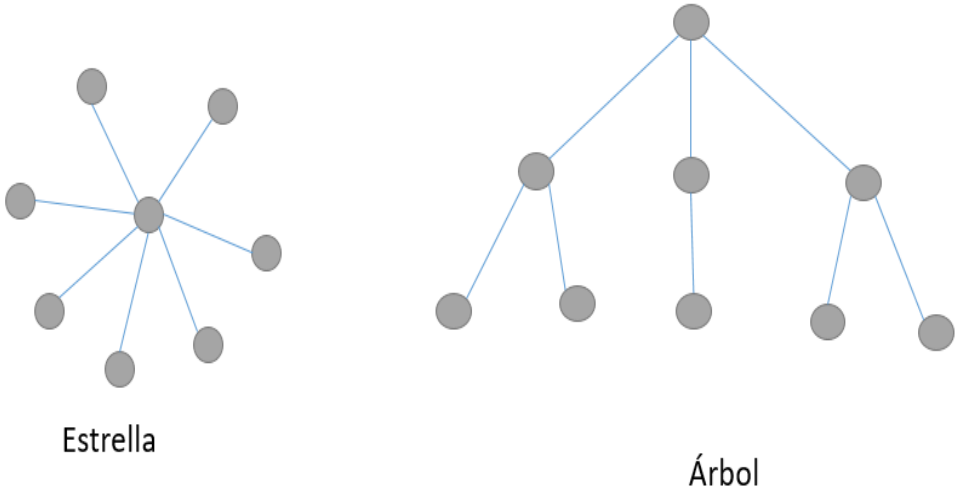

Figura 3. Tipologías de: Estrella y Árbol

3. MARCO REGULADOR Y MARCO SOCIO-ECONOMICO

3.1 Marco regulador

En el ámbito de la domótica todos los dispositivos deben cumplir una serie de reglamentos en España, siguiendo la normativa europea, como son:

- Reglamento Electrotécnico de Baja Tensión y su Guía de aplicación, [10].
- Reglamento Particular de la marca AENOR para Instalaciones de Sistemas Domóticos en Viviendas [11] y su Especificación [12].
- Documento Básico DB-HE "Ahorro de Energía", del Código Técnico de la Edificación [13].

En todo momento se debe de tener en cuenta estas normas en el desarrollo de cualquier aplicación en el ámbito de la domótica.

Según AENOR [12] hay que distinguir tres niveles de domotización de las viviendas:

- Nivel 1: Instalaciones con un nivel mínimo de dispositivos o aplicaciones domóticas con un mínimo de 13 puntos y tres aplicaciones.
- Nivel 2: En este caso la suma de puntos debe de ser de 30 puntos mínimo y tres aplicaciones.
- Nivel 3: En el que la suma mínima de puntos según las tablas de AENOR debe de ser 45 con 6 aplicaciones.

3.2 Marco socio-económico

La implementación de sistemas domóticos en la vivienda tiene más ventajas a parte de las evidentes de comodidad del usuario y ahorro energético puesto que también generan una revalorización de la vivienda debido a que un sistema domótico es un valor añadido a la hora de realizar la compraventa de una vivienda, añadiendo el factor de ahorro energético para obtener una mejor calificación energética de la misma.

Los costos de la implantación actual de los sistemas de domótica son muy elevados, en el caso de los dispositivos KNX [7], los cuales son uno de los sistemas de domótica más extendidos, cada dispositivo KNX [7] cuesta de media entre los 100€ y los 300€ a lo cual hay que sumar los costes de instalación de los mismos, existen otros sistemas como las luces inteligentes basadas en sistemas Zigbee [8] coste aproximado de 40€ por dispositivo.

En los últimos años la facturación del sector de los sistemas de control y domótica ha sufrido una evolución positiva considerable pasando de 37,8M€ en 2013 a 56,7M€ en 2017 como podemos observar en la *Figura 4*, siendo significativo este crecimiento entre los años 2016 y 2017 el cual supone un 12%.

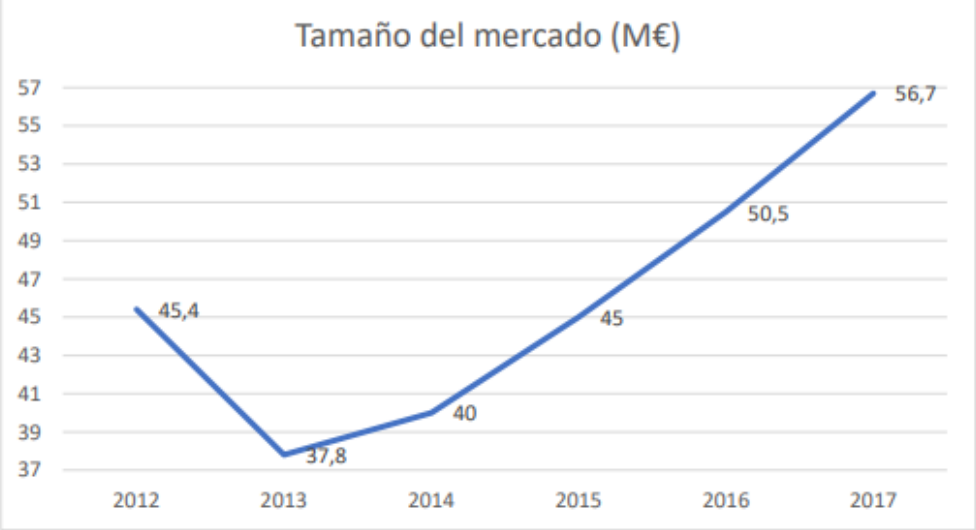

Figura 4. Facturación de los fabricantes de sistemas de control y automatización [14]

En la actualidad la inversión en I+D+i en el ámbito de la domótica ha aumentado considerablemente en los últimos años, lo cual producirá una reducción en los costes de fabricación de los sistemas domóticos.

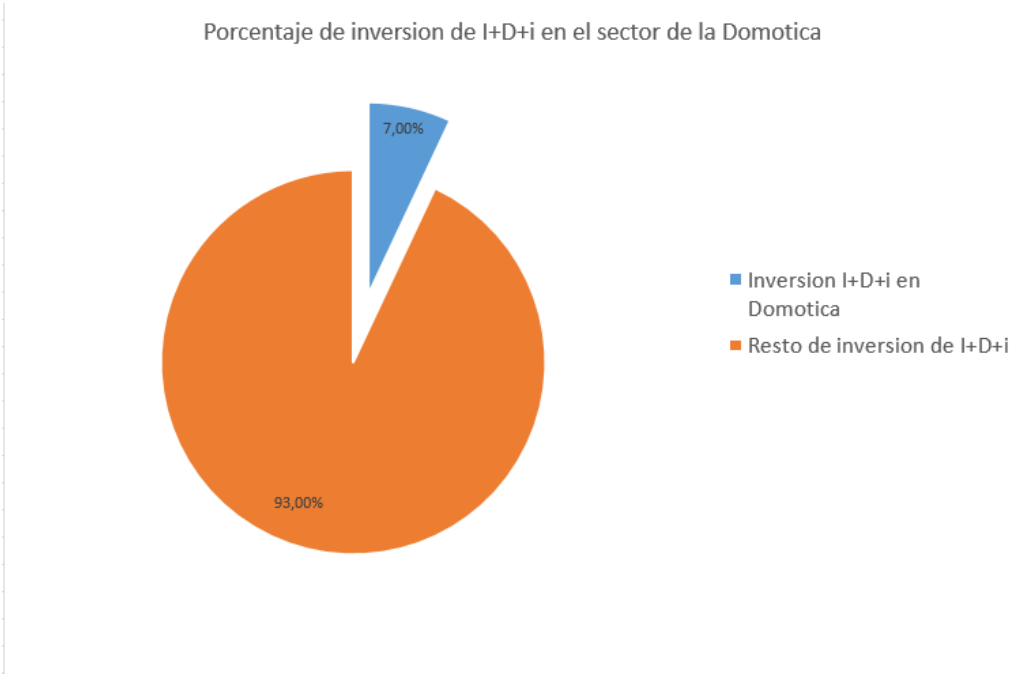

Figura 5. Porcentaje de inversión de I+D+i en el sector de la domótica

Si bien es cierto que la gran mayoría de las instalaciones de domótica se realizan en viviendas residenciales de obra nueva. Seguidas, pero con casi la mitad, están los edificios del sector terciario y con un pequeño porcentaje están las obras de rehabilitación de viviendas.

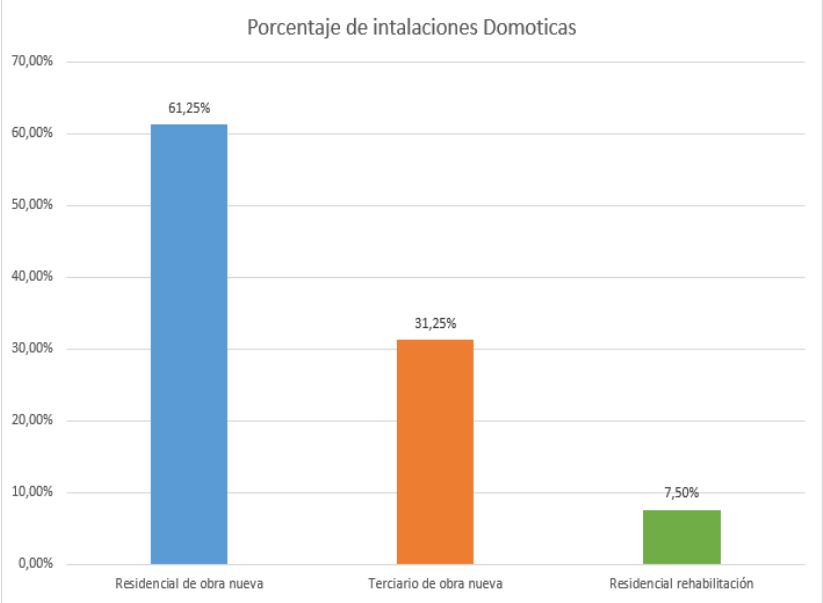

Figura 6. Porcentajes de instalaciones domóticas

El costo de instalación en viviendas de obra nueva tampoco es muy elevado dado que comprende entre un 1,6% y un 6,8% de media del presupuesto de la vivienda, dicho porcentaje comparado con el presupuesto que se suele asignar a la instalación eléctrica, el cual ronda de media el 3% del mismo.

Debemos tener en cuenta que en la actualidad casi todas las empresas del ámbito de la domótica trabajan con sistemas propietarios, es decir, el protocolo de comunicación entre los dispositivos domóticos ha sido desarrollado por la empresa para sus dispositivos, los cual dificulta la integración de distintos dispositivos de distintos fabricantes en la misma red dado que no sería posible que se comunicasen entre ellos ya que no se entenderían.

Si bien es cierto hay varios protocolos estandarizados y bastante extendidos, como son los comentados anteriormente, KNX [7], Zigbee [8] etc. Aún queda mucho para el ideal de comprar un dispositivo cualquiera y que se integre sin problemas en nuestra red.

También hay que comentar que se están dando grandes pasos en esta dirección dado que grandes marcas como puede ser Philips utilizan estos protocolos estándar para sus dispositivos domóticos, en el caso de Philips se ve en sus bombillas inteligentes, aunque también los podemos observar en los Google Home y los Amazon Echo que son capaces

de controlar sin problemas cualquier dispositivo que se comunique mediante el protocolo Zigbee.

Otro de los aspectos a tener en cuenta es que muchas de las empresas que utilizan estos protocolos sacan dispositivos que nos permitirían integrar los distintos protocolos en una sola red.

En el ámbito social los dispositivos domóticos son instalados principalmente a usuarios de menos de 34 años, en esta franja de edad también se encuentra un gran porcentaje de personas con intención de instalar dispositivos domóticos. En cambio, en las personas mayores de 50 años solamente un 13% de ellas tienen instalados dispositivos domóticos y un 39% no tiene intención de adquirirlos. De ello observamos que la edad es uno de los factores más importantes que marcan el consumo de productos domóticos como observamos en la *Figura 7.*

Figura 7. Domótica: Diferencias por tramos de edad [15]

A este entorno hay que añadir la proliferación de los asistentes virtuales de las principales empresas tecnológicas del mundo las cuales están compitiendo entre sí para quedarse con el mercado de los mismos, estos asistentes están integrados en sencillos aparatos dotados de micrófono y altavoz, pero también se pueden generar redes domóticas en las cuales pueden ejecutar acciones mediante comandos de voz.

Figura 8. Principales asistentes virtuales

Para finalizar, según la consultora Gartner Research [15], en 2020 habrá aproximadamente 20.800 millones de dispositivos conectados en todo el mundo, y según la consultora IDC [15] el mercado de los mismos alcanzara un valor 1,7 billones de dólares.

4. DISEÑO DEL SISTEMA DOMOTICO PROPUESTO

4.1 Ámbito de actuación

Dentro de este proyecto de domotización de los sistemas de una vivienda se encuentran distintos elementos, cada uno con su singularidad. Para afrontar las distintas necesidades nos valdremos de una serie de chips existentes en el mercado, a los que añadiremos los elementos necesarios en cada caso para adecuarlas a las funciones necesarias y alcanzar los objetivos de nuestro proyecto.

Se va a necesitar diseñar módulos para el control de la iluminación, tomas de corriente, persianas, calefacción, control de temperatura y el módulo de procesamiento central.

4.2. Tarjetas de control

Siguiendo los objetivos del presente proyecto se han seleccionado los componentes a utilizar en el mismo teniendo en cuenta el coste y la no necesidad de realizar una canalización u obra para la instalación, igualmente se ha buscado la posibilidad por la que los componentes puedan ser programados mediante librerías de código libre para minimizar los costes de desarrollo.

Las opciones en este campo son limitadas, al no querer realizar una nueva canalización para el sistema las posibilidades quedan reducidas al uso de ondas portadoras o señales de radio para la comunicación de los diferentes dispositivos, aunque si la vivienda también contiene una instalación de red ethernet también sería posible utilizarla, debido a no ser estándar en el mercado existente no vamos a tomarla en consideración.

De los dos métodos posibles, el de ondas portadoras nos obliga a sustituir todos los sistemas tradicionales eléctricos por sistemas con comunicación de ondas portadoras por lo que se ha decidido descartarlo por su elevado costo, lo cual produce que nos hayamos decantado por el uso de la radio frecuencia.

En el espectro de la radio frecuencia hay que tener mucho cuidado con el ancho de banda utilizado debido a la fuerte regulación vigente sobre el uso de las bandas de frecuencia, por ello se ha optado por desarrollar el sistema con una comunicación mediante el estándar Wifi dado el gran número de chips wifi de bajo costo y alta calidad a disposición que existen en el mercado que cumplen la regulación y normativa sobre las ondas de radio.

Para la elección del montaje de los módulos se han barajado distintas opciones entre las que se encontraban las tarjetas arduino MKR 100 o las diversas tarjetas arduino con distintos chips wifi incorporados, por ello lo primero era escoger el chip wifi adecuado.

Dentro del amplio mercado que este sector nos proporciona se ha escogido el chip ESP8266 el cual es un chip wifi de bajo costo y a la vez muy potente, el cual se puede adquirir en distintas tarjetas en función de las necesidades de nuestro sistema.

El chip ESP8266 incluye los siguientes elementos:

- Una pila TCP/IP completa
- Una CPU de 32bits a 80MHz
- Una RAM de instrucción de 64KB y una de datos de 96KB
- WIFI 802.11 b/g/n
- 16 pines GPIO
- 1 pin SPI I2C
- 1 conversor ADC de 10 bit

Este chip tiene la capacidad de generar una red propia, lo cual es muy interesante en el ámbito de la domótica permitiéndonos no saturar de dispositivos la red doméstica de la vivienda y realizar la instalación domótica en viviendas sin una red de wifi doméstica.

Tiene tres modos de funcionamiento el modo *"station"* el cual nos permite conectarnos a una red wifi, el modo *Access Point* (AP) genera nuestra propia red wifi y un modo mixto que nos permite tanto generar nuestra propia red como conectarnos a una red ya existente.

Se barajó la opción de utilizar este chip en conjunto con alguna tarjeta arduino pero al investigar más a fondo se encontraron una serie de tarjetas basadas en este chip las cuales tenían la ventaja de ser más potentes que las tarjetas arduino convencionales como la arduino uno y la arduino nano, eliminando el inconveniente de la comunicación entre el chip ESP8266 y la tarjeta arduino lo cual simplificaba mucho la programación de los módulos.

El único posible inconveniente del chip ESP8266 radica en que solo permite la conexión de 8 dispositivos de manera simultánea por lo cual no permitiría una amplia red de dispositivos domóticos.

Dentro de todas las tarjetas basadas en el chip ESP8266 en concreto las tarjetas utilizadas en el proyecto son: la NodeMCU y la ESP-01s, sería posible utilizar alguna otra de las tarjetas basadas en este chip, aunque las que se han observado óptimas para este proyecto son las anteriormente mencionadas dado su sencillez a la hora de conectarle los periféricos necesarios para el diseño de los módulos.

Cada una de estas tarjetas tiene distintas características a tener en cuenta, aunque ambas tarjetas están basadas en el mismo chip cuentan con numerosas diferencias entre sí, por lo cual las describiremos más detalladamente.

4.2.1 NodeMCU

Esta es una tarjeta de desarrollo orientada a IoT basada en el chip ESP8266, cuenta con una serie de ventajas frente a otras basadas en este chip, como son la incorporación de un chip conversor USB, permitiendo la conexión directa al ordenador para su programación, un regulador de tensión 5V/3,3V el cual nos permite alimentar la tarjeta mediante un puerto USB de ordenador, e incluye acceso a casi todos los pines del chip ESP8266.

La tarjeta la podemos observar en la *Figura 9.*

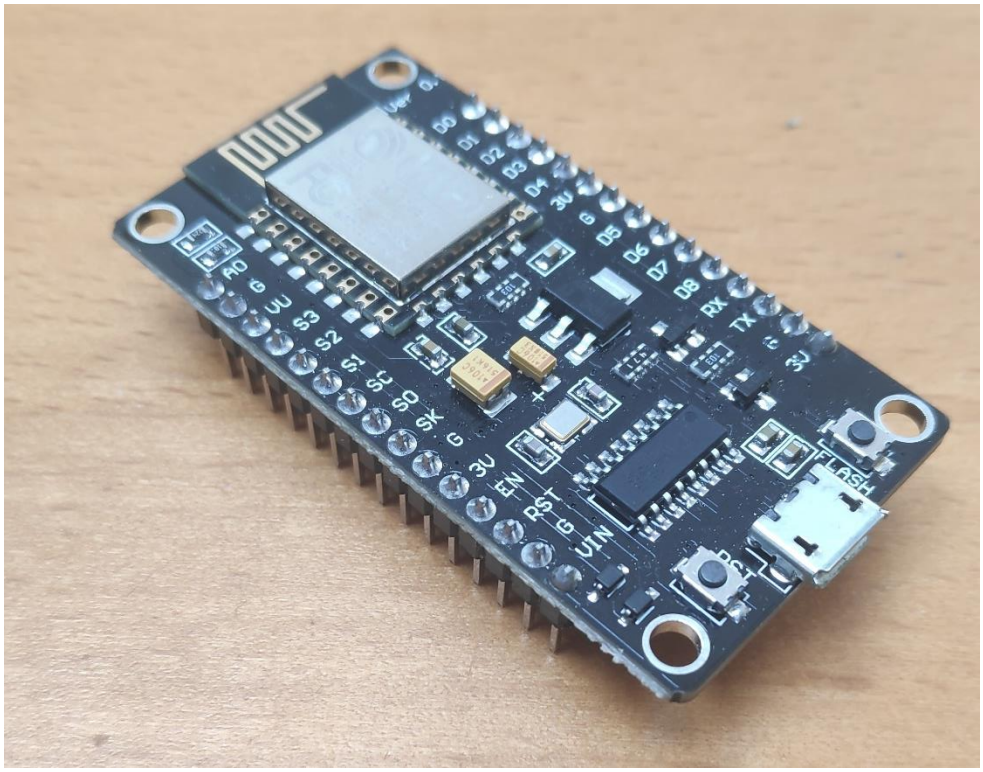

Figura 9. Tarjeta NodeMCU

La tarjeta NodeMCU cuenta con tres versiones, las cuales difieren en distintos aspectos no esenciales como son el tamaño de la placa, el chip de conversor USB, o el número de leds en la placa.

La versión de esta tarjeta que se va a utilizar para el proyecto es la NodeMCU v.3, la cual es la última versión de esta tarjeta, esta versión monta un conversor serial CH340G, cuenta con un pin de alimentación a 5V a parte de la posibilidad de alimentarla por el conector micro-USB, esta tarjeta monta el chip ESP8266 en una placa ESP-12E soldada

directamente a la NodeMCU, esta trajeta tiene accesibles los pines gpio 0-5, 9-10, 12-13 y 15-16; además cuenta con un botón de *Reset* y otro de *Flash*.

La tarjeta tiene unas dimensiones de 5,7 cm x 3 cm.

En esta tarjeta se instalarán el software del módulo central, del control de las persianas y de la adquisición de temperatura y humedad.

4.2.2 ESP-01s

Esta tarjeta también está basada en el chip ESP8266, aunque difiere de la anterior en el hecho de no ser una tarjeta de desarrollo.

Como podemos observar en la *Figura 10.*

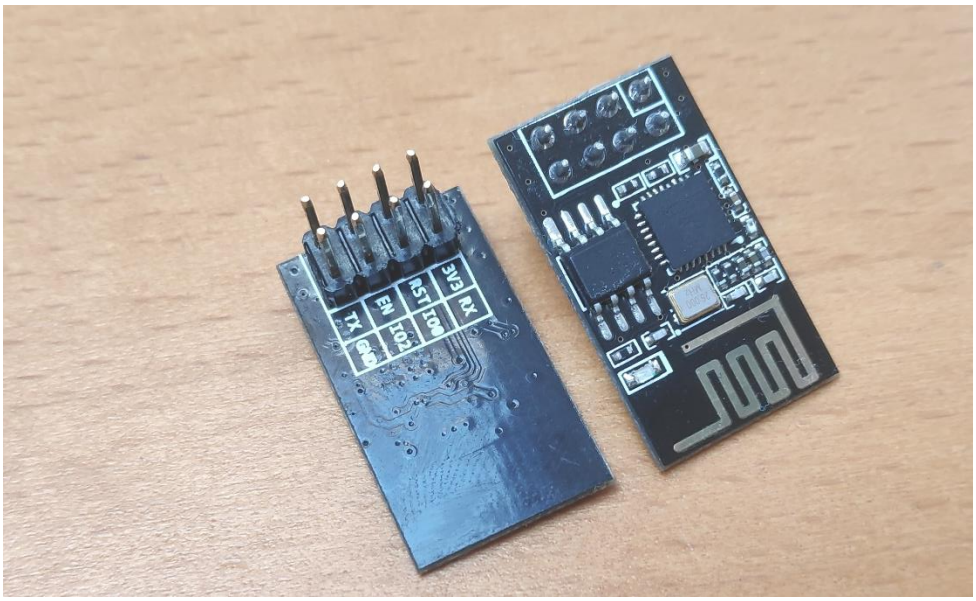

Figura 10. Tarjeta ESP-01s

Esta tarjeta no cuenta con un chip conversor USB por lo que se requerirá de uno para su programación, tampoco cuenta con un regulador de tensión por lo cual la alimentación de la tarjeta es de 3,3V con la cual hay que tener mucho cuidado ya que cualquier pico de tensión quemara la tarjeta.

Esta tarjeta cuenta con dos modelos el ESP-01 y el ESP-01s los cuales son prácticamente iguales y solo difieren en unas resistencias colocadas en los pines gpio, el tamaño de la memoria *Flash*, los leds de la tarjeta.

Esta tarjeta cuenta con 8 pines en total, dos pines gpio el 0 y el 2, un pin Rx otro Tx, un pin Vcc de 3,3V, un pin GND, un pin reset y un pin CH_PD para encender la tarjeta.

La tarjeta tiene unas dimensiones de 2.4 cm x 1,4 cm.

En esta tarjeta se instalará el software de los módulos relé para el control de luces, enchufes, puerta y calefacción.

4.3. Periféricos

En el presente proyecto se utilizarán una serie de componentes adicionales a las tarjetas anteriormente descritas para el correcto funcionamiento del mismo.

Estos periféricos constan de una serie de placas con un relé integrado las cuales son las ESP-01s relay v1.0, un sensor de temperatura y humedad que es el DHT22, un transformador HI-link HLK-PM01.

4.3.1. ESP-01s relay v1.0

Este periférico consta de un relé de 5V de activación, un regulador de tensión que baja los 5V de entrada a los 3,3V que utiliza el ESP-01s, un rack de pines hembra para conectar la tarjeta ESP-01s, el rack de pines hembra de la palca del relé viene ya preparado para la alimentación del ESP-01s, el circuito tiene una entrada para alimentar todo el conjunto a 5V y otra para conectar los cables al relé la cual tiene tres entradas para el común, el normalmente abierto y el normalmente cerrado.

La cual observamos en la *Figura 11.*

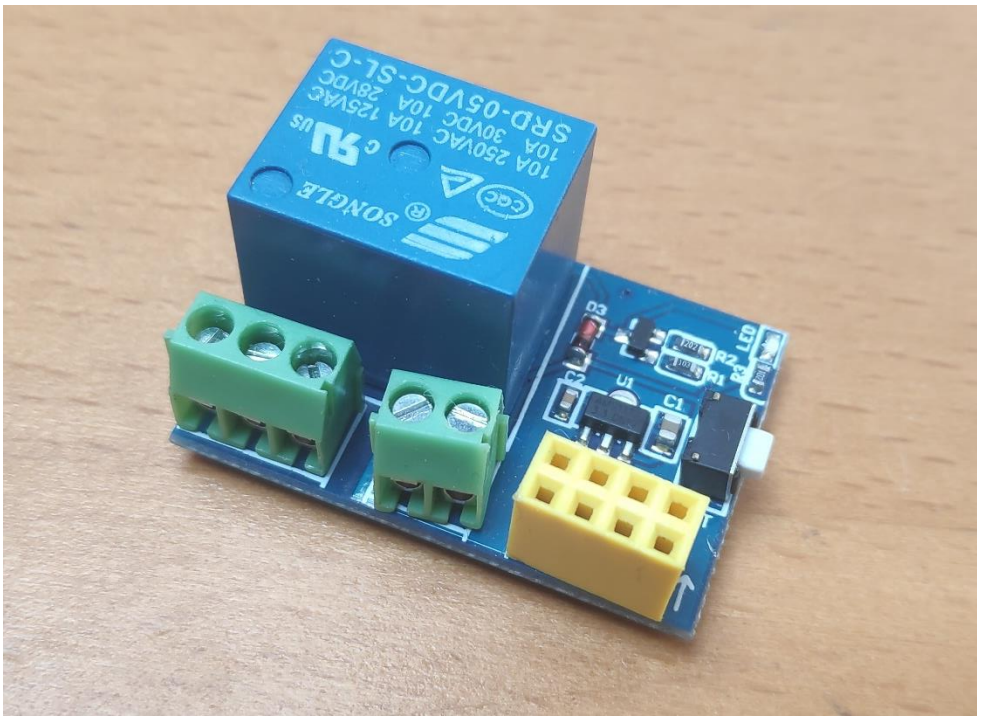

Figura 11. Placa ESP-01s Relay v1.0

Debido a un fallo del este dispositivo la tarjeta ESP-01s no arranca si está conectada en el rack de pines hembra de la palca del relé, este fallo surge de la conexión del pin gpio0 en el rack de pines hembra de la palca del relé del relé el cual se ha solucionado colocando un interruptor entre el pin gpio0 y la entrada en el rack de pines hembra de la palca del relé, y para que si se desconecta la alimentación sea sencillo el reinicio de la placa se instalara otro interruptor entre el pin de alimentación del ESP-01s y el rack de pines hembra de la palca del relé.

Esta placa tiene unas dimensiones de 3,7 cm x 2,5 cm x 1,7 cm.

4.3.2. DHT22

Este periférico consta de un sensor de temperatura y de un sensor de humedad, estos vienen montados en una placa para acondicionadora para su uso con mayor comodidad.

El cual se puede ver en la *Figura 12.*

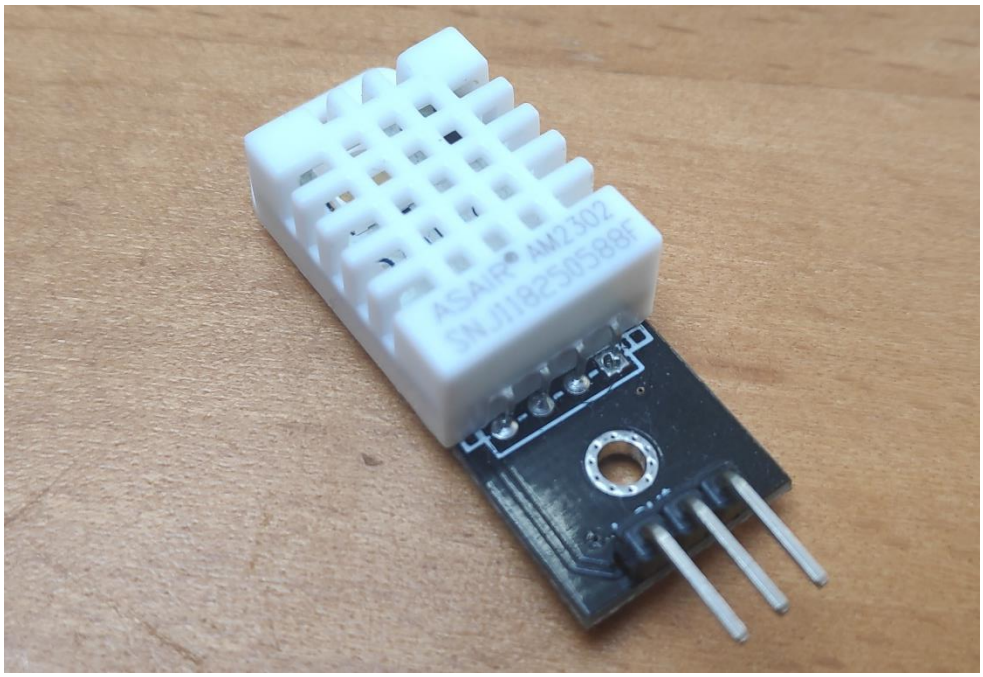

Figura 12. Sensor DHT22

Este sensor puede medir una temperatura de entre -40 y 125ºC con una precisión de 0,5ºC, mide la humedad entre el 0% y el 100% con una precisión de 2-5%, y tiene una frecuencia de muestreo de 2Hz.

Este periférico tiene unas dimensiones de 3,8 cm x 1,5 cm x 1 cm.

4.3.3. HI-link HLK-PM01

Este es un transformador de 3W con 100-240V AC a 50-60Hz de entrada y 5V DC y 0.6A de salida, este transformados lleva el sello CE pro lo que es un producto adecuado para su uso en Europa.

El cual se observa en la *Figura 13.*

Figura 13. Transformador Hi-Link.

Este transformador tiene unas dimensiones de 3,4 cm x 2cm x1,5 cm.

4.4. Módulos

Los módulos que cuentan con una tarjeta ESP-01s y una placa relé necesitan de un conexionado especial como se comentó en el aparatado de las placas relé, este conexionado se detalla en la *Figura 14.*

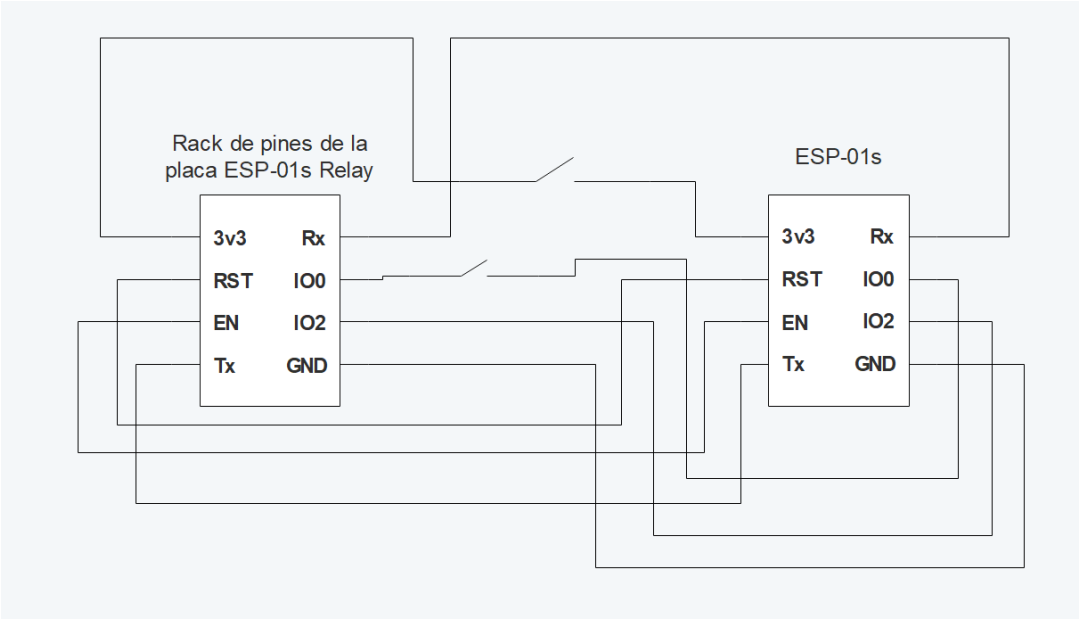

Figura 14. Esquema de conexionado entre el ESP-01s y la placa relé

Como se observa en la *Figura 14*, todos los conexionados se realizan directamente salvo las conexiones del pin de 3v3 y de gpio0 los cuales pasan por un interruptor antes, el interruptor de los 3v3 no es necesario debido a que el único pin que presenta problemas es el gpio0 pero si se da la situación de un corte de luz la placa no iniciaría de forma correcta al volver la luz y para reiniciar la placa con el pin gpio0 desconectado habría que desconectar el módulo de la alimentación lo cual es más complicado y peligroso por lo cual se ha instalado en interruptor en el pin 3v3 para poder apagar y volver a encender el módulo de manera sencilla y segura para el usuario.

A continuación, se detalla el diseño de cada uno de los módulos de manera independiente y como conectarlos a la alimentación y a los sensores y actuadores que utilizan.

4.4.1. Iluminación

El módulo de iluminación está compuesto por una tarjeta ESP-01s, una placa con relé, también se montará junto con un transformador para poder alimentarla adecuadamente.

Para que este módulo funcione de manera correcta ha de estar conectado como lo estaría el primer conmutador del circuito de iluminación de la habitación para que de esta manera sea posible seguir usando la instalación de la manera tradicional como se observa en la *Figura 15*.

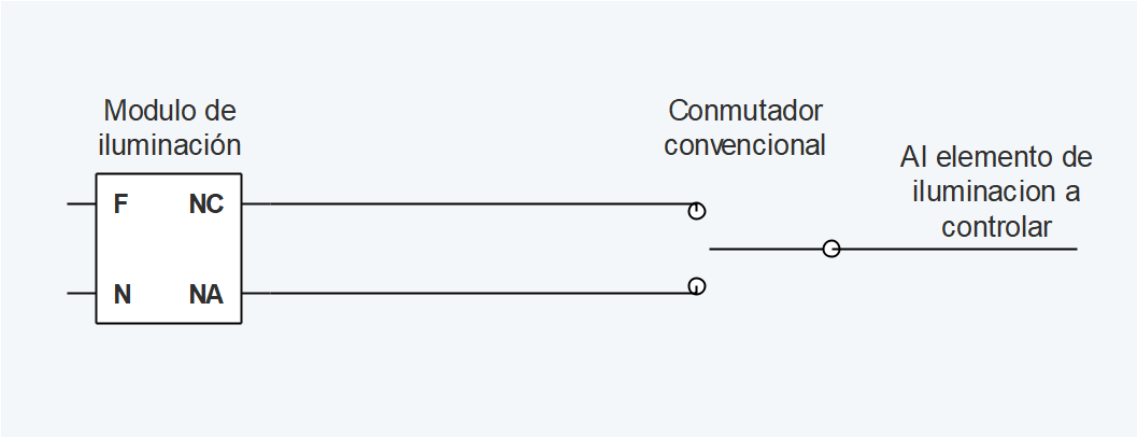

Figura 15. Conexión del módulo de iluminación a la red eléctrica

También será necesario que el segundo cable de la red eléctrica llegue al módulo para poder alimentarlo mediante su conexión en el terminal del transformador, la conexión de este transformador se detalla en la *Figura 16.*

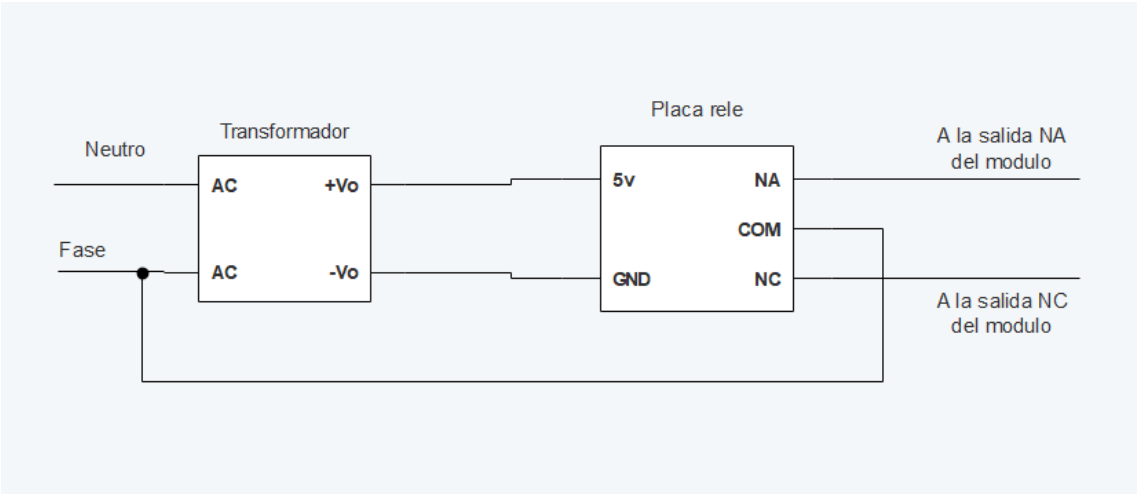

Figura 16. Esquema de conexionado del transformador a la placa relé

Como se ve en la *Figura 16*, la red eléctrica se conecta directamente al transformador y se aprovecha esa misma conexión para sacar la conexión al relé y que no sea necesario realizarla por fuera del módulo.

4.4.2 Enchufes

El módulo para el control de los enchufes es el mismo que el de iluminación, solo se diferencia en el método de conexión y las conexiones de los cables dentro del módulo.

Para conectar este módulo hay que conectar ambos cables de la red eléctrica a los conectores de transformador y sacar los dos cables al enchufe del conector del módulo para el enchufe como se observa en la *Figura 17*.

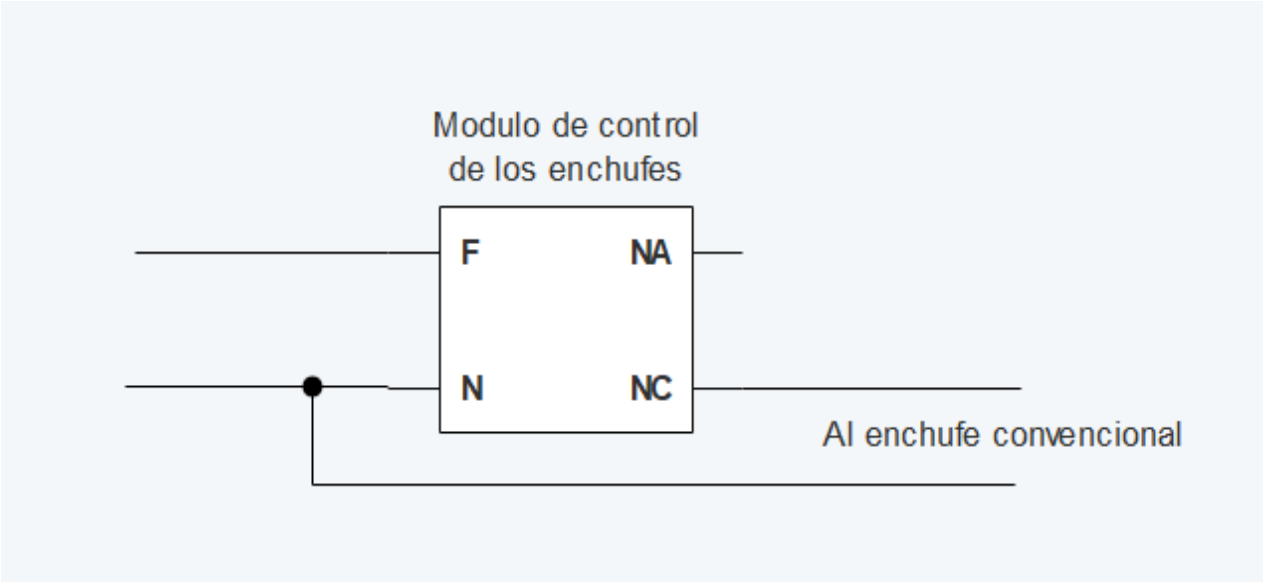

Figura 17. Conexión del módulo de enchufes a la red eléctrica

4.4.3. Persianas

Este módulo está compuesto por una tarjeta NodeMCU, dos relés para el control del motor de la persiana y los conectores para el motor y los sensores.

Para el correcto funcionamiento de este módulo hay que conectar las salidas del motor del módulo directamente al motor junto con las conexiones anteriores al motor y el común de los relés al cable antes de la conexión con el pulsador y los sensores final de carrera a sus respectivos conectores en el módulo, todo ello se observa en la *Figura 18*.

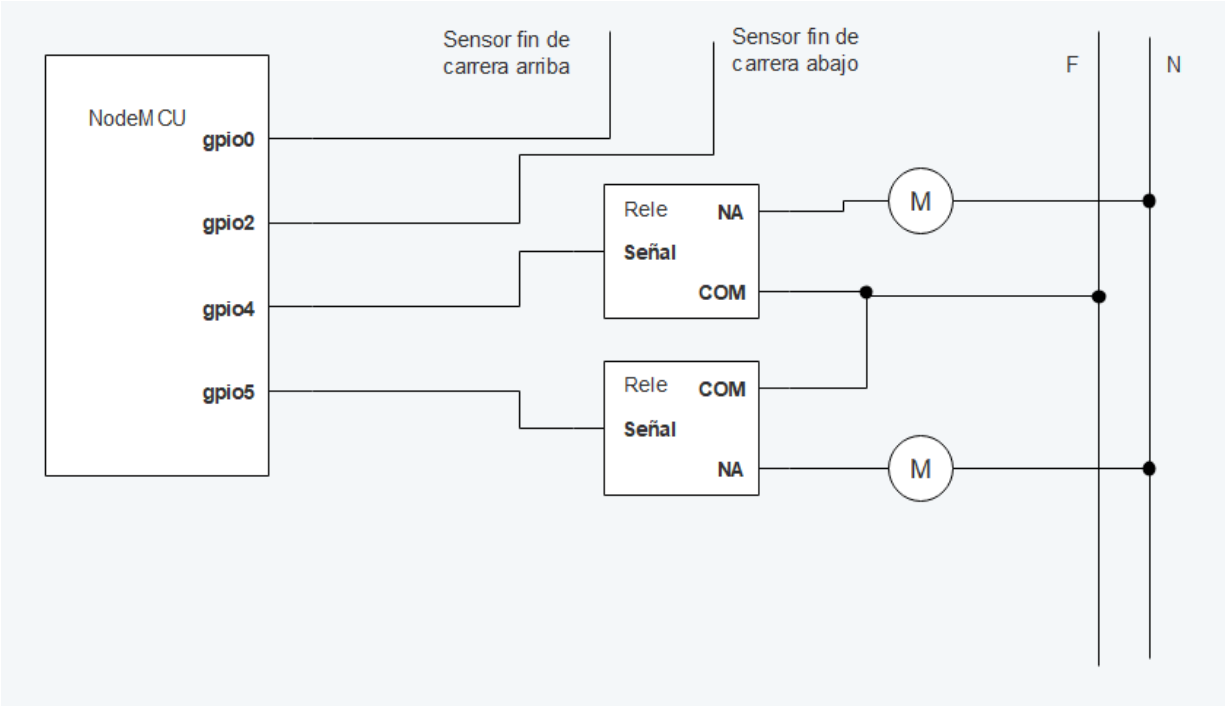

Figura 18. Esquema de conexión del módulo de control de persianas

Para alimentar este módulo solo es necesario conectar un cargador de móvil con salida micro-USB al puerto micro-USB del módulo.

4.4.4. Control de calefacción

Este módulo está compuesto por un conjunto de tarjeta ESP-01s, una placa relé y un transformador, la conexión del módulo ESP-01s con la placa relé se ha detallado anteriormente en la *Figura 14*.

Para conectar este módulo es necesario llevar los dos cables de la red eléctrica para alimentar el módulo, también se tendrá que conectar la salida del relé a el cable que activa la caldera y el cable antes del relé del termostato a la entrada del relé como se detalla en la *Figura 19*.

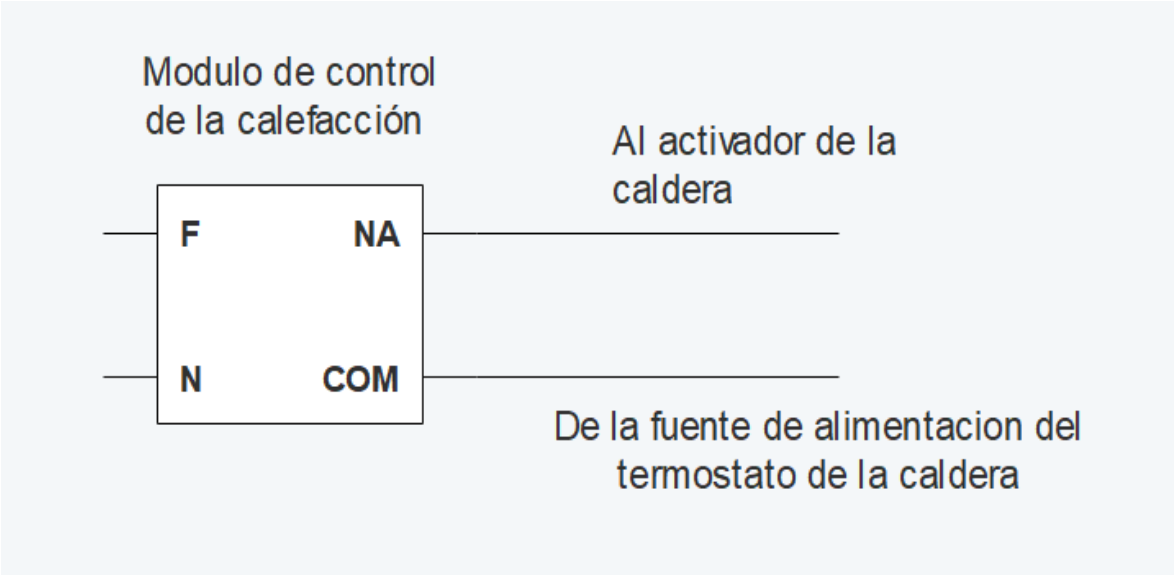

Figura 19. Esquema de conexión del módulo de control de la caldera

4.4.5. Control de apertura de la puerta

Este módulo está diseñado para funcionar en la apertura de un telefonillo, aunque puede servir para cualquier sistema de apertura electrónica.

Este módulo está compuesto por un conjunto de tarjeta ESP-01s una placa relé y un transformador, la conexión entre la placa ESP-01s y la placa relé esta detallada en la *Figura 14*.

Para que este módulo pueda funcionar junto con el sistema tradicional es necesario conectar el común del relé al cable de alimentación del telefonillo y la salida del relé al cable que va a la puerta para abrirla como se detalla en la *Figura 20*.

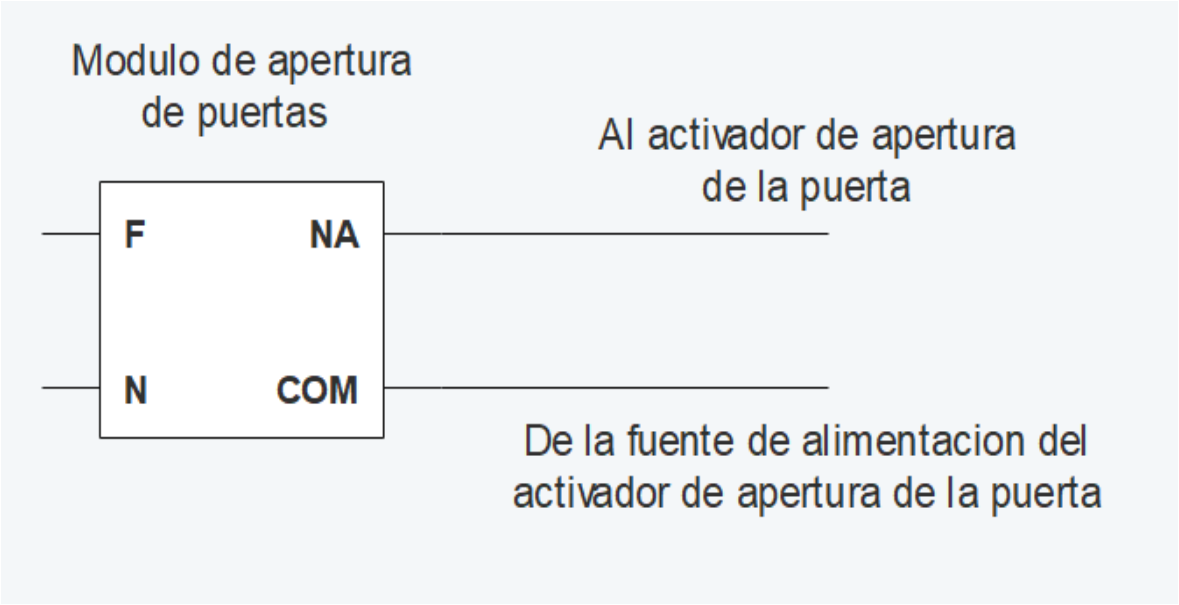

Figura 20. Esquema de conexión del módulo de apertura de la puerta

4.4.6. Adquisición de temperatura y humedad

Este módulo está compuesto por una tarjeta NodeMCU y un sensor DHT22.

Este módulo no necesita de ningún conexionado a la red eléctrica a parte de la alimentación, pero para una óptima medición es aconsejable situar el módulo a una altura media, que no esté situado muy cerca de ventanas, puertas, radiadores o sistemas de aire acondicionado o calefacción.

La conexión entre la placa NodeMCU y el sensor DHT22 consta solo de la alimentación del sensor y de la conexión del pin de datos del sensor como se detalla en la *Figura 21*.

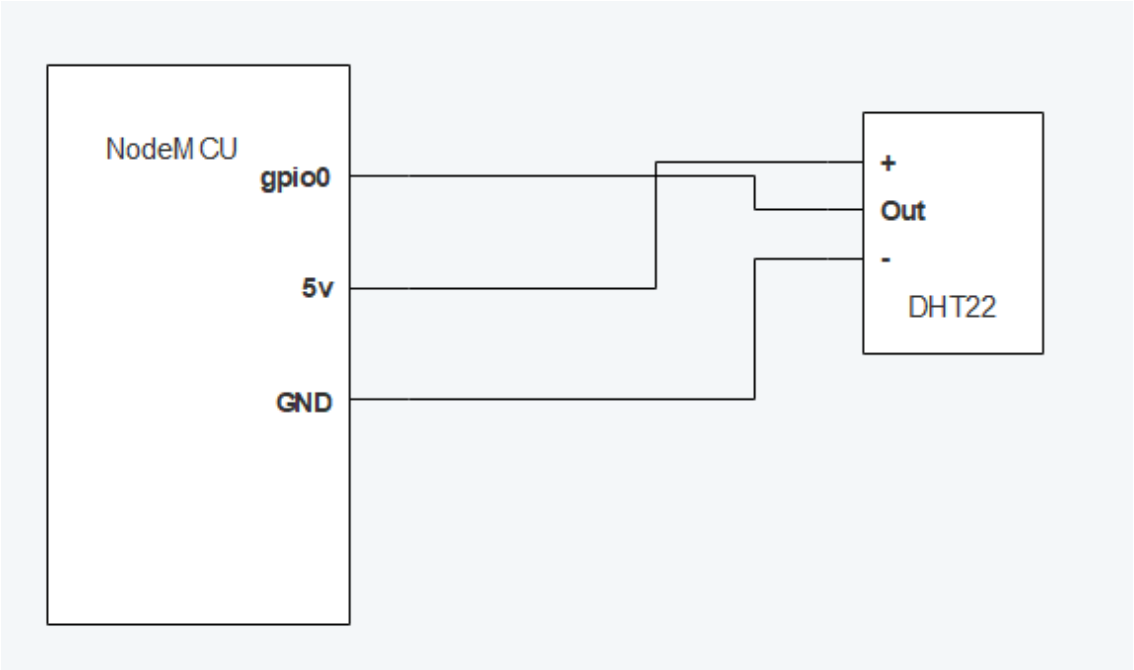

Figura 21. Esquema de conexión entre el NodeMCU y el DHT22

Para alimentar este módulo solo es necesario conectar un cargador de móvil con salida micro-USB al puerto micro-USB del módulo.

4.4.7. Control

Este módulo está compuesto únicamente por una tarjeta NodeMCU por lo que no necesita ninguna conexión.

No necesita ninguna conexión externa a parte de la alimentación.

Sí sería aconsejable no colocar el módulo cerca de estructuras metálicas ni de *routers* ya que estos elementos pueden afectar a la intensidad de la señal de la placa.

Para alimentar este módulo solo es necesario conectar un cargador de móvil con salida micro-USB al puerto micro-USB del módulo.

4.5. Otras consideraciones

En todos los módulos compuestos por tarjetas ESP-01s dicha tarjeta puede ser sustituida por una tarjeta NodeMCU realizando los consiguientes cambios de conexiones, pero los

módulos con tarjetas NodeMCU no se puede cambiar esta tarjeta pos una ESP-01s dado que esta última no deja acceso a los pines necesarios para el funcionamiento de los módulos, la única excepción seria el módulo de control, pero para una alimentación más sencilla se ha escogido la placa NodeMCU.

5. IMPLEMENTACION DEL SISTEMA DOMOTICO

5.1. Introducción

En este parte del proyecto se procede a explicar cómo se ha desarrollado el proyecto a nivel de software.

Para programar los dispositivos se ha utilizado el Android IDE.

El chip ESP8266 tiene dos métodos de programación, el primero sería programarlo mediante comandos AT, este método tiene sus limitaciones y es complejo la programación de grandes programas, por ello existe otro método, este otro método es mediante la librería ESP8266Wifi.h la cual mediante una serie de funciones facilita la configuración y el uso de este chip, esta librería es *Open Source*.

En la configuración del chip se nos permite ponerlo en un modo que genera nuestra propia red wifi y conectarnos a una red wifi ya existente, pero este modo tiene el problema que no nos permite quedarnos esperando a la respuesta que recibimos de otro módulo cuando le mandamos una petición debido a que salta el Whatchdog dado que si está esperando a la respuesta del otro módulo no puede atender a las peticiones que le llegan desde la red wifi a la que está conectada.

Para el desarrollo de estos módulos se ha optado por una topología de estrella con el módulo de control en el centro, se ha tenido en cuenta que la mejor topología para estos casos es la topología de malla pero dado que las librerías del ESP8266 no están pensadas para funcionar con esta topología no nos es posible conectarlo de esta manera, al igual que pasa con la topología de anillo, se podría realizar la topología en árbol pero debido a la imposibilidad del chip de esperar la respuesta de un dispositivo conectado al mientras está conectado a su vez a otra red ya se generada por otro chip o por un *router* no podríamos hacer más que un nivel del árbol lo cual es una tipología de estrella, esta tipología la podemos observar en la *Figura 22*.

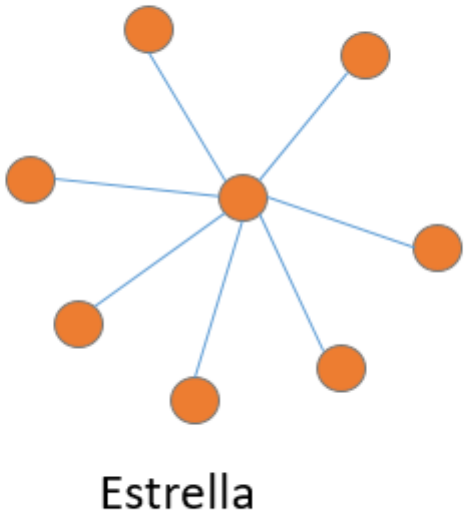

Figura 22. Tipología de estrella

5.2. Módulos

Todos los módulos tienen una parte en común, la cual observamos en la *Figura 23.*

```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>

String ssid = "Domotica";
String password="";
String tipodisp="1";//0=Desconocido,1=rele,2=dht,3=persianas,4=telefonillo,5=termostato,6=Control
String nombre="Rele";
```

Figura 23. Librerías y variables comunes

Esta parte común consiste en las librerías del chip ESP8266 que facilitan la programación del chip estas librerías son “ESP8266WiFi.h” y “ESP8266WebServer.h”.

La librería “ESP8266WiFi.h” tiene las funciones para la programación del chip, algunas de estas funciones son las usadas para establecer el modo de funcionamiento del chip,

establecer la conexión a una red wifi o generar nuestra red wifi, todo esto en función del modo que estemos utilizando.

La librería “ESP8266WebServer.h” nos facilita el crear un servidor en el módulo el cual pueda atender peticiones HTTP de los dispositivos, y realizar acciones diferenciadas en función de la petición recibida.

También tenemos el nombre de SSID a la que se conectara el módulo o de la red que generara el módulo y su contraseña.

Luego se establece el tipo de dispositivo que esta codificado mediante un entero.

Y para terminar establecemos el nombre de la placa para que sea fácilmente identificada por el usuario.

Otra de las partes del código común en todos los módulos es la que se puede observar en la *Figura 24.*

```
11  ESP8266WebServer server(80);
```

Figura 24. Configuración del servidor

Esta línea del código establecemos el puerto a través del cual se comunicará el servidor.

5.2.1. Iluminación

El código de este módulo generara un servidor web en la placa para el cual se programarán las acciones del módulo en función de la petición HTTP que le llegue al módulo, también se conectara de forma automática a la red creada por el módulo de control.

Como observamos en la *Figura 25*, el código principal del módulo está ubicado en la función *“setup”* donde se configura el modo del módulo al modo *“station”* para que se conecte a la red generada por el módulo de control y también establece las funciones que se ejecutaran dependiendo de la petición recibida por el servidor.

```
83  void setup() {
84    Serial.begin(115200);
85    WiFi.mode(WIFI_STA);
86    if(WiFi.begin(ssid)){
87      Serial.println("conectado");
88    }else{
89      Serial.println("Fallo de conecxion");
90    }
91    server.on("/",conecxion);
92    server.on("/tipo",sendtipo);
93    server.on("/gpio",rele);
94    server.on("/setnombre",setnombre);
95    server.on("/getnombre",getnombre);
96    server.on("/conf",conf);
97    server.on("/guardar_conf",guardar);
98    server.begin();
99    pinMode(Pin_rele,OUTPUT);
100 }
101
102 void loop() {
103   server.handleClient();
104 }
```

Figura 25. Función "setup" y "loop" del módulo de Iluminación

Dentro de estas funciones hay un grupo de ellas que son comunes para todos los módulos salvo para el módulo de control.

La primera de las funciones es la de conexión la cual se ejecuta cuando se accede directamente a la placa sin mandarle una petición concreta, esta función no realiza ninguna acción como se ve en *Figura 26*, pero es necesaria para que realice las acciones correctas cuando recibe el resto de peticiones.

```
12  void conecxion(){
13  }
```

Figura 26. Función "conexión"

La función "*sendtipo"* devuelve el tipo del módulo al módulo que le haya mandado la petición "/tipo", para lo cual utilizamos la función "*send"* como se ve en la *Figura 27*, del servidor la cual nos permite establecer la respuesta que enviara el servidor, con la cual enviamos el tipo del módulo el cual se codifica como se observa en la *Figura 28*.

```
18    server.send(200,"text/html",aux);
```

Figura 27. Función "send" del servidor

```
16    aux+="Datos=Tipo=";
```

Figura 28. Codificación de los datos en la función "sendtipo"

La función "*setnombre*" cambia el nombre del módulo cuando se le manda el nuevo nombre el módulo desde el módulo de control con la petición "/*setnombre*", para ello utilizamos la función "*arg"* la cual vemos en la *Figura 29*, la cual nos permite recuperar la *String* del nombre de la petición HTTP mediante la etiqueta del *String*.

```
31    nombre=server.arg("nombre");
```

Figura 29. Función "arg" del servidor

La función "*getnombre*" envía el nombre de este módulo al módulo de control cuando le llega la petición "/*getnombre*", esta función tiene el mismo funcionamiento a la "*sendtipo"*, utilizamos la función "*send"* vista en la *Figura 27* para mandar el nombre al módulo de control, codificado como podemos observar en la *Figura 30*.

```
36    aux="Datos=Nombre=";
```

Figura 30. Codificación de los datos de la función "getnombre"

La función "*conf*" manda una respuesta en forma de código HTML, cuando recibe la petición "/*conf*" para que desde un navegador puedas cambiar la configuración del módulo, esta configuración consiste en cambiar la red a la que se conecta el módulo en la página web generada se nos pedirá la introducción del SSID y de la contraseña de la red a la que se quiere conectar el módulo, para mandar la página web simplemente introducimos el código HTML en la función "*send*" del servidor como observamos en la *Figura 31*.

```
void conf(){  // Funcion que genera la pagina web para cambiar la red a la que se conecta
  Serial.println("config");
  server.send(200,"text/html","<!DOCTYPE html>"
                              "<html>"
                              "<head>"
                              "<title>Configuración</title>"
                              "<meta charset='UTF-8'>"
                              "</head>"
                              "<body>"
                              "</form>"
                              "<form action='guardar_conf' method='get'>"
                              "SSID:<br><br>"
                              "<input class='inputl' name='ssid' type='text'><br>"
                              "PASSWORD:<br><br>"
                              "<input class='inputl' name='pass' type='text'><br><br>"
                              "<input class='boton' type='submit' value='GUARDAR'/><br><br>"
                              "</form>"
                              "</body>"
                              "</html>");
}
```

Figura 31. Función "conf"

La función *"guardar"* guarda los parámetros introducidos en la página web generada por la función "*conf"*, debido a que el botón de la página web generada manda la petición "*/guardar_conf"*, esta función utiliza la función *"arg"* para recoger los parámetros de la red introducidos en la página web como ya se ha visto en la *Figura 29*, y dependiendo de los parámetros que han sido introducidos y se conecta de una manera o de otra mediante la función "*begin*" tal y como observamos en las *Figura 32* y *Figura 33*.

```
if (WiFi.begin(ssid)){
```

Figura 32. Función "begin" SSID

```
if (WiFi.begin(ssid,password)){
```

Figura 33. Función "begin" SSID y contraseña

Estas son todas las funciones comunes a todos los módulos a excepción del módulo de control por lo cual se obviarán en la explicación en los próximos módulos.

Ahora se procederá a describir las funciones propias de este módulo.

La función "*rele*" es la única función propia del módulo, esta función se ejecuta cuando le llega una petición "*/gpio*", esta función cambia el estado en el que se encuentra el relé es decir si el relé está abierto lo cierra y viceversa, para mandar la señal de activación y

desactivación del relé se utiliza la función "*digitalWrite*" como podemos observar en la *Figura 34.*

```
23      digitalWrite(Pin_rele,HIGH);
```

Figura 34. Función "digitalWrite"

En la función "*loop"* solamente hay una línea de código y consta de la función "*HandleClient"* la cual realiza la función de estar escuchando a ver cuándo le llega una petición al servidor para poder atenderla como hemos visto en la *Figura 25*.

El código completo de este módulo se puede observar en el *Anexo código* en *Código completo Módulo Iluminación y Enchufes*

5.2.2. Enchufes

Este módulo lleva el mismo código que el de iluminación solo difiere en la conexión de los cables como ya se ha comentado anteriormente, por lo cual no se procederá a explicarlo para no extender la explicación en demasía, al igual que el código se puede observar en el mismo anexo que el módulo anterior.

5.2.3. Persianas

El código de este módulo generara un servidor web en la placa para el cual se programarán las acciones del módulo en función de la petición HTTP que le llegue al módulo, también se conectara de forma automática a la red creada por el módulo de control.

Como observamos en la *Figura 35* el código principal del módulo está ubicado en la función "*setup*" donde se configura el modo del módulo al modo "*station*" para que se conecte a la red generada por el módulo de control y también establece las funciones que se ejecutaran dependiendo de la petición recibida por el servidor.

```
95  void setup() {
96    Serial.begin(115200);
97    WiFi.mode(WIFI_STA);
98    if(WiFi.begin(ssid)){
99      Serial.println("conectado");
100   }else{
101     Serial.println("Fallo de conecxion");
102   }
103   server.on("/",conecxion);
104   server.on("/tipo",sendtipo);
105   server.on("/setnombre",setnombre);
106   server.on("/getnombre",getnombre);
107   server.on("/conf",conf);
108   server.on("/guardar_conf",guardar);
109   server.on("/subir", subir);
110   server.on("/bajar", bajar);
111   server.on("/parar", parar);
112   server.begin();
113 }
114
115 void loop() {
116    server.handleClient();
117    boolean arriba=digitalRead(pinSensorArriba);
118    boolean abajo=digitalRead(pinSensorAbajo);
119    if(arriba==true||abajo==true){
120       digitalWrite(pinSubir,LOW);
121       digitalWrite(pinBajar,LOW);
122    }
123 }
```

Figura 35. Funciones "setup" y "loop" del módulo de persianas

En este módulo se repite la configuración del modo del chip ESP8266, así como las funciones conexión *Figura 26*, "*sendtipo*" *Figura 27 y Figura 28*, *"setnombre" Figura 29*, *"getnombre" Figura 30*, "*conf*" *Figura 31*, "*guardar*" *Figura 32* y *Figura 33*.

Ahora procederé a explicar aquellas partes del código que son propias de este módulo, para comenzar en la función *"loop"*, la cual observamos en la *Figura 35*, tenemos las condiciones necesarias para la parada del motor de la persiana cuando la persiana llega al principio o al final de su recorrido.

La función subir colocara los relés en la posición adecuada para que la persiana empiece a subir, esto se realiza colocando un relé abierto y el otro cerrado de tal manera que el motor se active girando en el sentido necesario para subir la ventana, para abrir o cerrar

los relés se utiliza la misma función que se usaba en el módulo relé es decir la función "*digitalWrite*", como pudimos observar en la *Figura 34*.

La función bajar funciona de igual manera que la función subir, pero invierte la posición de los relés para que el motor en vez de girar en el sentido de subida lo haga en el de bajada.

La función parar colocara los relés en la posición de desconexión para parar el motor se pare sea cual sea el estado en el que se encuentre, subiendo, bajando o parado.

El código completo de este módulo se puede encontrar en el *Anexo código* en el apartado de *Código completo Módulo Persianas.*

5.2.4. Control de calefacción

El código de este módulo generara un servidor web en la placa para el cual se programarán las acciones del módulo en función de la petición HTTP que le llegue al módulo, también se conectara de forma automática a la red creada por el módulo de control.

```
82  void setup() {
83     Serial.begin(115200);
84    WiFi.mode(WIFI_STA);
85    if(WiFi.begin(ssid)){
86      Serial.println("conectado");
87    }else{
88      Serial.println("Fallo de conecxion");
89    }
90    server.on("/",conecxion);
91    server.on("/tipo",sendtipo);
92    server.on("/setnombre",setnombre);
93    server.on("/getnombre",getnombre);
94    server.on("/conf",conf);
95    server.on("/guardar_conf",guardar);
96    server.on("/encender",encender);
97    server.on("/apagar",apagar);
98    server.begin();
99    pinMode(Pin_rele,OUTPUT);
100 }
101
102 void loop() {
103   server.handleClient();
104 }
```

Figura 36. Funciones "setup" y "loop" del módulo de control de la calefacción

Como se observa en la *Figura 36,* el código principal del módulo está ubicado en la función *"setup"* donde se configura el modo del módulo al modo *Station* para que se conecte a la red generada por el módulo de control y también establece las funciones que se ejecutaran dependiendo de la petición recibida por el servidor.

En este módulo se repite la configuración del modo del chip ESP8266, así como las funciones conexión *Figura 26, "sendtipo" Figura 27 y Figura 28, "setnombre" Figura 29, "getnombre" Figura 30, "conf" Figura 31, "guardar" Figura 32* y *Figura 33*.

A continuación, se procederá a la explicación de las funciones propias de este módulo, la primera de ellas es la función encender la cual activa el relé para encender la caldera mediante la función "*digitalWrite"* como ya se ha visto anteriormente.

La otra función propia de este módulo es la función "*apagar*" la cual realiza la función inversa a la anterior comentada, es decir coloca el relé en la posición de desconexión para apagar la caldera mediante la función "*digitalWrite"* como se ha explicado en módulos anteriores.

Podemos observar el código completo del módulo en el *Anexo código* en el apartado de *Código completo Módulo Control de calefacción.*

5.2.5. Control de apertura de puerta

El código de este módulo generara un servidor web en la placa para el cual se programarán las acciones del módulo en función de la petición HTTP que le llegue al módulo, también se conectara de forma automática a la red creada por el módulo de control.

Como se observa en la *Figura 37* y *Figura 38* el código principal del módulo está ubicado en la función *"setup"* donde se configura el modo del módulo al modo *Station* para que se conecte a la red generada por el módulo de control y también establece las funciones que se ejecutaran dependiendo de la petición recibida por el servidor.

```
103 void setup() {
104   Serial.begin(115200);
105   WiFi.mode(WIFI_STA);
106   if(WiFi.begin(ssid)){
107     Serial.println("conectado");
108   }else{
109     Serial.println("Fallo de conecxion");
110   }
111   server.on("/",conecxion);
112   server.on("/tipo",sendtipo);
113   server.on("/gpio", abrir);
114   server.on("/setnombre",setnombre);
115   server.on("/getnombre",getnombre);
116   server.on("/conf",conf);
117   server.on("/guardar_conf",guardar);
118   server.on("/tiempo",tiempo);
119   server.on("/guardar_tiempo",guardar_tiempo);
120   server.begin();
121 }
```

Figura 37. Función "setup" del módulo de apertura de puerta

```
122
123 void loop() {
124   server.handleClient();
125 }
```

Figura 38. Función "loop" del módulo de apertura de puerta

En este módulo se repite la configuración del modo del chip ESP8266, así como las funciones conexión *Figura 26, "sendtipo" Figura 27 y Figura 28, "setnombre" Figura 29, "getnombre" Figura 30, "conf" Figura 31, "guardar" Figura 32* y *Figura 33*.

A continuación se explicaran las funciones propias del módulo, la primera de ellas es la función "*abrir*" la cual activa el relé durante el tiempo establecido para abrir la puerta y pasado dicho tiempo lo desactiva, esto se puede observar en la Figura 39.

```
75 void abrir(){ // Funcion para abrir el telefonillo
76   Serial.println("abrir");
77   digitalWrite(Pin_rele,HIGH);
78   delay((tiempoabrir*1000));
79   digitalWrite(Pin_rele,LOW);
80 }
```

Figura 39. Función "abrir"

A continuación, se observa la función "*tiempo*", la cual se ejecuta cuando el módulo recibe una petición "/*tiempo*", esta función devuelve una página web en la que se nos pide establecer el nuevo valor del tiempo que va a estar el módulo activando la apertura de la puerta, esta función es igual a la función *"conf"* vista en la *Figura 31*, en lo único en lo que se diferencian es la página web generada.

La última función exclusiva de este módulo es la función "*guardar_tiempo*" la cual se ejecuta cuando enviamos al módulo el dato del nuevo tiempo y utiliza la función *"arg"* para obtener el nuevo parámetro del tiempo de la petición y guardar el nuevo tiempo.

Todo el código de este módulo se encuentra en el *Anexo código* en el apartado de *Código completo Módulo Apertura de puerta.*

5.2.6. Adquisición de temperatura y humedad

El código de este módulo generara un servidor web en la placa para el cual se programarán las acciones del módulo en función de la petición HTTP que le llegue al módulo, también se conectara de forma automática a la red creada por el módulo de control.

En este módulo se repite la configuración del modo del chip ESP8266, así como las funciones conexión *Figura 26, "sendtipo" Figura 27 y Figura 28, "setnombre" Figura 29, "getnombre" Figura 30, "conf" Figura 31, "guardar" Figura 32* y *Figura 33*.

Las funciones propias de este módulo son la función "*datos*" la cual adquiere los datos de temperatura y humedad del sensor DHT y lo manda como respuesta a la petición "/*dht*", el código de la función se puede observar en la *Figura 40* en esta figura se observa cómo se obtienen los valores de temperatura y humedad del sensor mediante las funciones propias del sensor "*readTemperature"* y "*readHumidity"*, a continuación se calcula el valor de la sensación térmica mediante otra de las funciones propias del sensor la cual es la función "*computeHeatIndex"* y por último se envían los datos de vuelta mediante la función "*send*" como se ha observado anteriormente.

```
82 void datos(){ // Funcion para mandar los datos de temperatura y humedad al control
83   String aux="";
84   do{
85     h = dht.readHumidity();
86     t = dht.readTemperature();
87     if (isnan(h) || isnan(t)) {
88       Serial.println("Failed to read from DHT sensor!");
89     }
90   }while(isnan(h) || isnan(t));
91   float hic = dht.computeHeatIndex(t, h, false);
92   aux+="Datos=Temp=";
93   aux+=t;
94   aux+="Hum=";
95   aux+=h;
96   aux+="TempAp=";
97   aux+=hic;
98   server.send(200,"text/html",aux);
99 }
```

Figura 40. Función "datos"

El código de este módulo se encuentra en el *Anexo código* en el apartado de *Código completo Módulo de Adquisición de temperatura y humedad*.

5.2.7. Control

El código de este módulo generara un servidor web en la placa para el cual se programarán las acciones del módulo en función de la petición HTTP que le llegue al módulo, este módulo al ser el módulo de control generara la red a la cual se conectaran el resto de los módulos y al cual nos conectaremos para controlar el sistema completo.

Como observamos en la *Figura 41*, el código principal del módulo está ubicado en la función *"setup"* donde se configura el modo del módulo al modo *Access Point* para que genere la red a la que se conectaran el resto de los módulos y también establece las funciones que se ejecutaran dependiendo de la petición recibida por el servidor.

```
277 void setup() {
278   Serial.begin(115200);
279   WiFi.mode(WIFI_AP);
280   if (WiFi.softAP(ssid)){
281     Serial.println("Ha generado la red");
282   }else{
283     Serial.println("No se ha generado la red");
284   }
285   server.on("/",conecxion);
286   server.on("/general",general);
287   server.on("/conf",conf);
288   server.on("/guardar_conf",configuracion);
289   server.on("/gettiponombre",gettiposnombre);
290   server.on(funcion[0],data1);
291   server.on(funcion[1],data2);
292   server.on(funcion[2],data3);
293   server.on(funcion[3],data4);
294   server.on(funcion[4],data5);
295   server.on(funcion[5],data6);
296   server.on(funcion[6],data7);
297   server.on(funcion[7],data8);
298   server.on("/cambiar_nombre",cambiarnombre);
299   server.on("/nombres",nombre);
300   server.on("/dht",dht);
301   server.on("/telefonillo",telefonillo);
302   server.on("/subir",subir_persiana);
303   server.on("/bajar",bajar_persiana);
304   server.on("/parar",parar_persiana);
305   server.on("/encender",encender);
306   server.on("/apagar",apagar);
307   server.begin();
308 }
309 void loop() {
310   server.handleClient();
311 }
```

Figura 41. Funciones "setup" y "loop" del módulo de control

En este módulo se repite la configuración del modo del chip ESP8266, así como las funciones conexión *Figura 26*, *"conf" Figura 31*, *"guardar" Figura 32* y *Figura 33*.

A continuación, se procederá a explicar las funciones propias de este módulo, para comenzar todas las funciones que se ejecutan cuando se realizan acciones en las páginas web para control del sistema se llama a la función "*paginawebprincipal"* la cual envía la página web principal del sistema, más adelante explicare esta función más a fondo.

La primera función "*gettiposnombre*" la cual se ejecuta cuando recibe la petición "/*gettiponombre*" y llama a tres funciones la función "*gettipo"*, la función "*getnombres"* y la función "*paginawebprincipal*", las cuales se explicarán posteriormente.

Las funciones "*data1-8*" sirven para controlar los módulos relé, constan de la llamada a la función "*conectrele*" y a la función "*paginawebprincipal*" como se ve en la *Figura 42*, la única diferencia entre las distintas funciones data es la dirección IP a la cual se mandara la orden.

```
55  // Las funciones data1-8 sirven para controlar los reles
56  void data1(){
57    conectrele(0);
58    paginawebprincipal();
59  }
```

Figura 42. Función "data1"

La función "*cambiarnombre*" generara una página web cuando el módulo de control reciba la petición "/*cambiar_nombre*", en dicha página web se nos pedirá que introduzcamos los nuevos nombres de los módulos, si dejamos el hueco en blanco, el nombre de ese módulo no será cambiado, para que solo nos pida los nombres de dispositivos conectados se realiza el código mediante un bucle "*For*" como se ve en la *Figura 43*, y a continuación se manda mediante la función "*send*" como se ha visto anteriormente.

```
99  for(int a=0;a<WiFi.softAPgetStationNum();a++){
100   c+=nombres[a];
101   c+=":<br>";
102   c+="<input class='input1' name='nombre";
103   c+=(a-1);
104   c+="' type='text'><br>";
105 }
106 c+="<input class='boton' type='submit' value='GUARDAR'/><br><br>";
107 c+="</form>";
108 c+="</body>";
109 c+="</html>";
110 server.send(200,"text/html",c);
111 }
```

Figura 43. Función "cambiarnombre"

La función "*nombre*" recoge los nombres introducidos en la página web generada por la función "*cambiarnombre*" mediante la función *"arg"* para recoger los nombres de la petición HTTP de la misma manera de lo comentado anteriormente con la única excepción de su introducción en un bucle "*For*" para realizar el tratamiento de todos los datos, como se observa en la *Figura 44.*

```
112 void nombre(){ // Funcion para guardar los nuevos nombres y mandarlos a las placas
113   for (int a=0;a<WiFi.softAPgetStationNum();a++){
114     String c,aux;
115     c+="nombre";
116     c+=(a-1);
117     aux=server.arg(c);
118     if (aux!=""&&aux!=" "){
119       nombres[a]=aux;
120       setnombre(a);
121     }
122   }
123   paginawebprincipal();
124 }
```

Figura 44. Función "nombre"

La siguiente función es "*dht*" la cual se ejecuta cuando el módulo recibe una petición "/*dht*", esta función llama a la función "*conectdht*" para obtener los datos de temperatura humedad y temperatura aparente del módulo DHT y genera una página web para poder mostrarlos, para la conexión se utiliza un bucle "*For*" para la búsqueda del módulo DHT, como se observa en la *Figura 45.*

```
129   for(int a=0;a<WiFi.softAPgetStationNum();a++){
130     if(tipo[a]==2){
131       conectdht(a);
132       delay(100);
```

Figura 45. Función "dht"

La función "*telefonillo*" se conecta con el módulo del telefonillo mediante la función "*conectrele*" la búsqueda se realiza de la misma manera que se ha comentado anteriormente, y se utiliza la función "*conectrele*" debido a la gran semejanza de este módulo con los módulos relé no es necesario mandarle una petición diferente, todo esto se observa en la *Figura 46.*

```
159 void telefonillo(){ // Funcion para controlar el telefonillo
160   for(int a=0;a<WiFi.softAPgetStationNum();a++){
161     if(tipo[a]==4){
162       conectrele(a);
163       paginawebprincipal();
164     }
165   }
166 }
```

Figura 46. Función "telefonillo"

A continuación, se procederá a la explicación de las funciones correspondientes al control de las persianas.

La función "*subir_persiana*" conecta con el módulo de control de las persianas mandándole la petición correspondiente, se busca la dirección del módulo de la misma manera a la comentada en las funciones anteriores.

La función "*bajar_persiana*" manda la petición correspondiente al módulo de control de la persiana para que coloque los relés de la manera adecuada para bajar la persiana, la búsqueda es igual a la ya comentada.

La función "*parar_persiana*" manda la petición adecuada al módulo de control de las persianas para desactivar los relés y que el motor de la persiana se pare parando de mover la persiana.

Las siguientes funciones son las necesarias para controlar el módulo de la caldera.

La función "*encender*" envía la petición necesaria al módulo de control de la caldera para activar el relé y encender la caldera.

La función "*apagar*" envía la petición necesaria al módulo de control de la calefacción para desactivar el relé y así apagar la caldera.

Ya han sido explicadas todas las funciones a las cuales se llaman en función de las peticiones HTTP que recibe el módulo, a continuación, procederé a explicar las funciones auxiliares que se utilizan en este módulo.

La primera de ellas es la función "*paginawebprincipal*" la cual envía como respuesta a una petición la página web de inicio de nuestro sistema, esta página web se realiza con código HTML y puede ser recibida por cualquier navegador, la página web consta de un apartado para cada módulo conectado e identificado del sistema, en estos apartados se observara el nombre del módulo así como los botones necesarios para su control, si un módulo no ha sido identificado no aparecerá en esta página web así como tampoco aparecen los móviles u ordenadores conectados al módulo. A continuación, se detallarán

los botones para: cambiar los nombres de los módulos, cambiar los parámetros de la red que genera y obtener los tipos y los nombres de los módulos conectados.

La siguiente función es la función "*gettipo*" la cual manda la petición correspondiente, para la obtención del tipo del módulo en la respuesta, esta petición se manda a todos los dispositivos conectados al módulo de control, para la conexión se utiliza la función "*connect*" a la cual se le pasa la dirección IP del dispositivo al cual nos queremos conectar y a continuación se utiliza la función "*println*" en el *client* para mandar la petición correspondiente, para la recepción de la respuesta se usa un bucle que escucha todo lo que manda el dispositivo mientras la conexión permanece activa y una vez recibida la finalización de la trasmisión el cliente se desconecta y se procede al tratamiento de los datos el cual consiste en buscar en el *String* recibido los datos requeridos, todo ello se puede observar en la *Figura 47*.

```
421 void gettipo(){ // Funcion para pedir a las placas su tipo
422   Serial.println("Gettipo");
423    for(int a=0;a<WiFi.softAPgetStationNum();a++){
424       Serial.println("conect");
425       String connection=ips[a];
426       String aux,line="";
427       aux="HOST: ";
428       aux+=connection;
429       WiFiClient client;
430       Serial.println(connection);
431       if(client.connect(connection,80)<0){
432         Serial.println("fallo en el envio");
433       }else{
434         client.println("GET /tipo HTTP/1.0");
435         client.println(aux);
436         Serial.println("enviado");
437         while (client.connected()){
438           if (client.available()){
439             line += client.readStringUntil('\n');
440             Serial.print("1 ");
441             Serial.println(line);
442           }
443         }
444          int pos=line.indexOf("Datos=");
445          String line2=line.substring(pos+6);
446          Serial.print("2");
447          Serial.println(line2);
448          pos=line2.indexOf("Tipo=");
449          tipo[a]=line2.substring(pos+5).toInt();
450         client.stop();
451       }
452    }
453 }
```

Figura 47. Función "gettipo"

La función "*conectrele*" manda la petición correspondiente a los módulos relé para que cambien su estado esto se realiza de la misma manera a la comentada en la función anterior pero sin esperar ninguna respuesta, como se observa en la *Figura 48.*

```
454 void conectrele(int a){ // Funcion que conecta con el rele en la posicion que se te ha pasado
455   Serial.println("Conect rele");
456   String connection=ips[a];
457   String aux;
458   aux="HOST: ";
459   aux+=connection;
460   WiFiClient client;
461   Serial.println(connection);
462   if(client.connect(connection,80)<0){
463     Serial.println("fallo en el envio");
464   }else{
465     client.println("GET /gpio HTTP/1.0");
466     client.println(aux);
467     Serial.println("enviado");
468     client.stop();
469   }
470 }
```

Figura 48. Función "conectrele"

La función "*conectdht*" manda la petición correspondiente al módulo DHT para recibir los datos de temperatura, humedad y temperatura aparente, esto lo realiza de la misma manera que se ha comentado en las funciones anteriores, conectando con el módulo correspondiente y esperando la respuesta de dicho módulo.

La función *"setnombre"* manda la petición correspondiente a los módulos para cambiarles el nombre del módulo, una vez ha sido introducido el nuevo nombre, para ello se conecta al dispositivo correspondiente de igual manera a la comentada anteriormente la única diferencia radica en que en la petición se le añade el nombre del módulo para que dicho módulo pueda recogerlo mediante la función *"arg"*, como se observa en la Figura 49, en esta función tampoco se espera a la respuesta del módulo dado que no es necesaria.

```
516     aux2+="GET /setnombre?nombre=";
517     aux2+=nombres[a];
518     aux2+=" HTTP/1.0";
519     client.println(aux2);
520     client.println(aux);
521     Serial.println("enviado");
522     client.stop();
523   }
```

Figura 49. Envío de los nuevos nombres

La función *"getnombre"* manda la petición correspondiente al módulo para la obtención en la respuesta del módulo el nombre de dicho módulo para ello se realiza la conexión de

la misma manera a la comentada anteriormente y se reciben los datos de igual manera que en la función "*gettipo*".

El código completo de este módulo así como todo el código HTML para generar las diversas páginas web utilizadas por el módulo está recogido en el *Anexo código* en el apartado de *Código completo Módulo Control*, en este anexo es interesante observar el código de la función "*paginawebprincipal*" dado que por su amplia extensión no se han introducido fragmentos en la memoria.

6. RESULTADOS Y EJEMPLO DE FUNCIONAMIENTO

6.1 Introducción

La solución obtenida en el presente proyecto está compuesta por 7 módulos distintos, los cuales son:

- Iluminación
- Enchufes
- Persianas
- Control de calefacción
- Apertura de puerta
- Temperatura y humedad
- Control

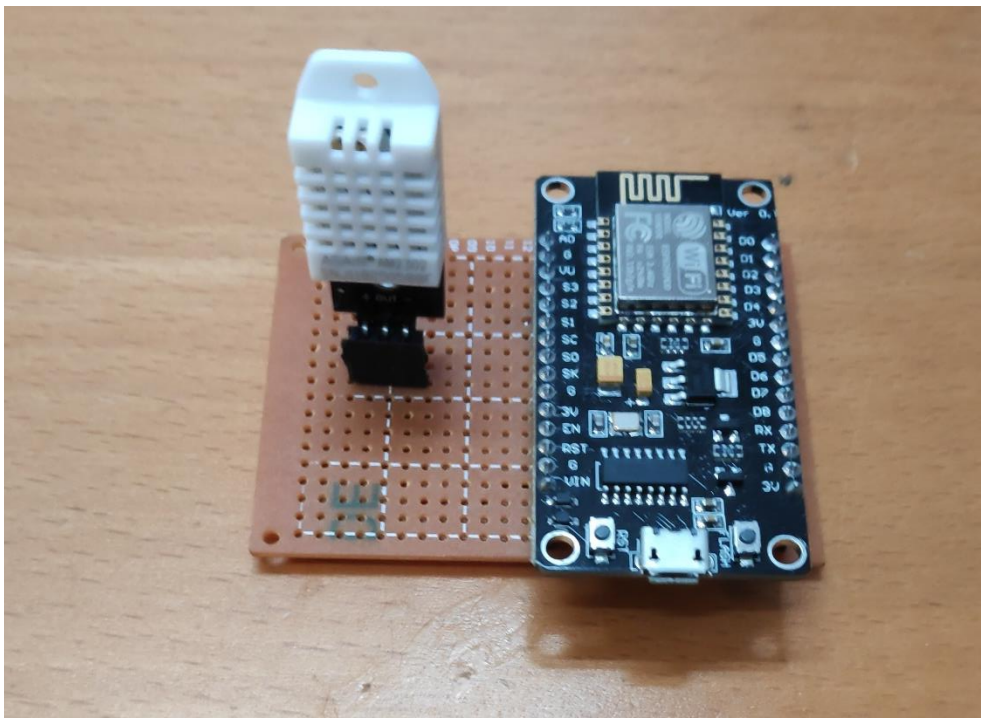

Figura 50. Módulo de Adquisición de temperatura y humedad

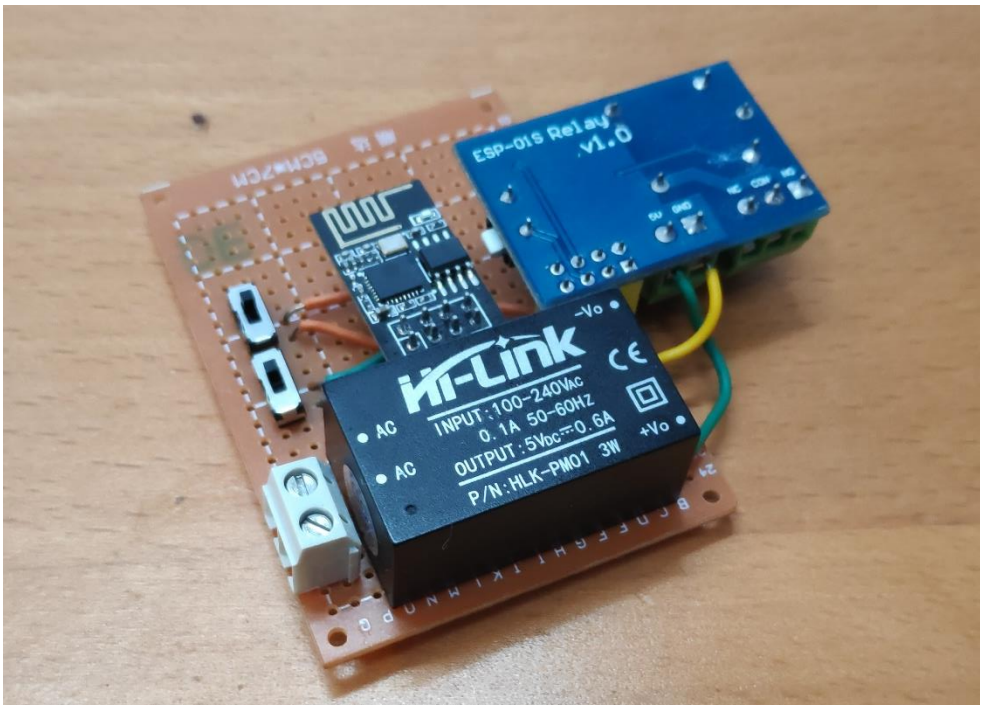

Figura 51. Módulo relé

Cada uno de los módulos tiene un funcionamiento distinto como se ha observado en los capítulos *4.* DISEÑO DEL SISTEMA DOMOTICO PROPUESTO y *5.* IMPLEMENTACION DEL SISTEMA DOMOTICO

Como lo visto en esos capítulos el módulo de control es el principal módulo del sistema dado que es el encargado de la coordinación y conexión del resto de módulos, a este módulo de control también será al cual nos conectemos desde un navegador ya se desde el móvil o desde un ordenador.

El módulo de control puede soportar un máximo de 8 conexiones debido a la capacidad del chip ESP8266. Se conectarán 6 dispositivos; un módulo de cada tipo al control, otro de los dispositivos posibles se deja libre para poderte conectar desde un navegador para su uso dejando un único hueco libre para otro módulo extra. Los únicos módulos que pueden ir duplicados son los de iluminación y enchufes permitiendo conectar a nuestra red domótica un módulo extra de este tipo.

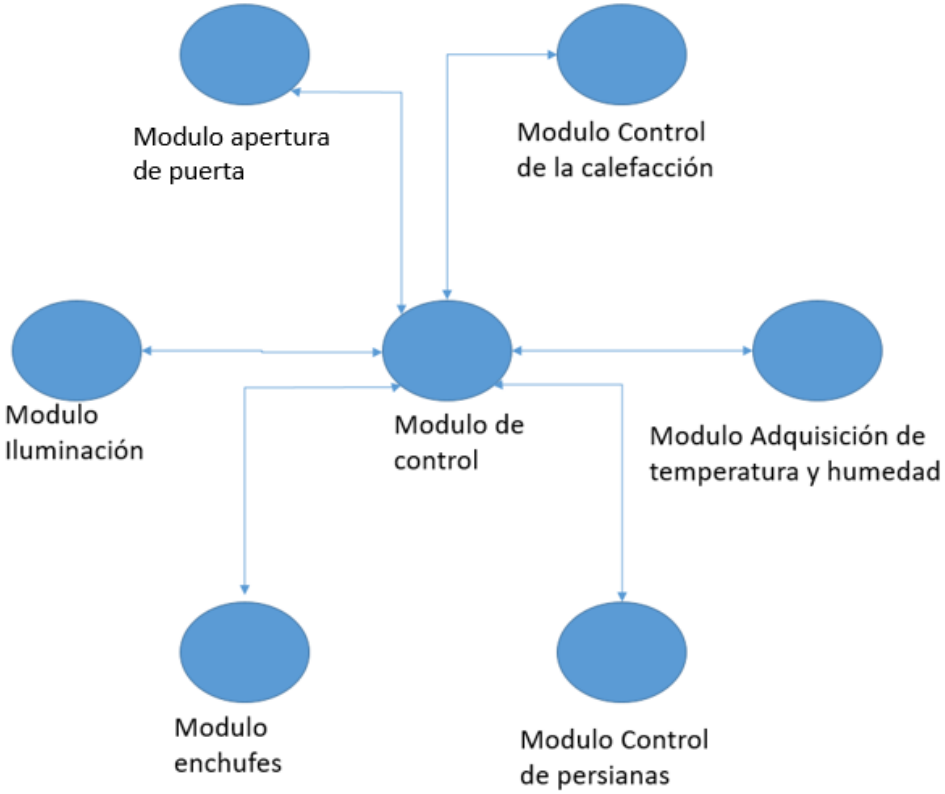

Figura 52. Esquema de conexión de todos los módulos al módulo de control

Como se puede observar en la *Figura 52*, cada módulo del sistema se conecta al módulo de control pero no entre si dando lugar a una tipología de estrella.

Ningún módulo es imprescindible para el funcionamiento de nuestro sistema como se puede ver en la *Figura 53*. Se pueden conectar varias combinaciones de módulos, aunque si bien es cierto necesitamos el módulo de control para poder configurar de manera independiente el resto de módulos y habría que mandarle directamente las peticiones específicas de las acciones a realizar desde el navegador por lo cual es recomendable tener un módulo de control en nuestra red.

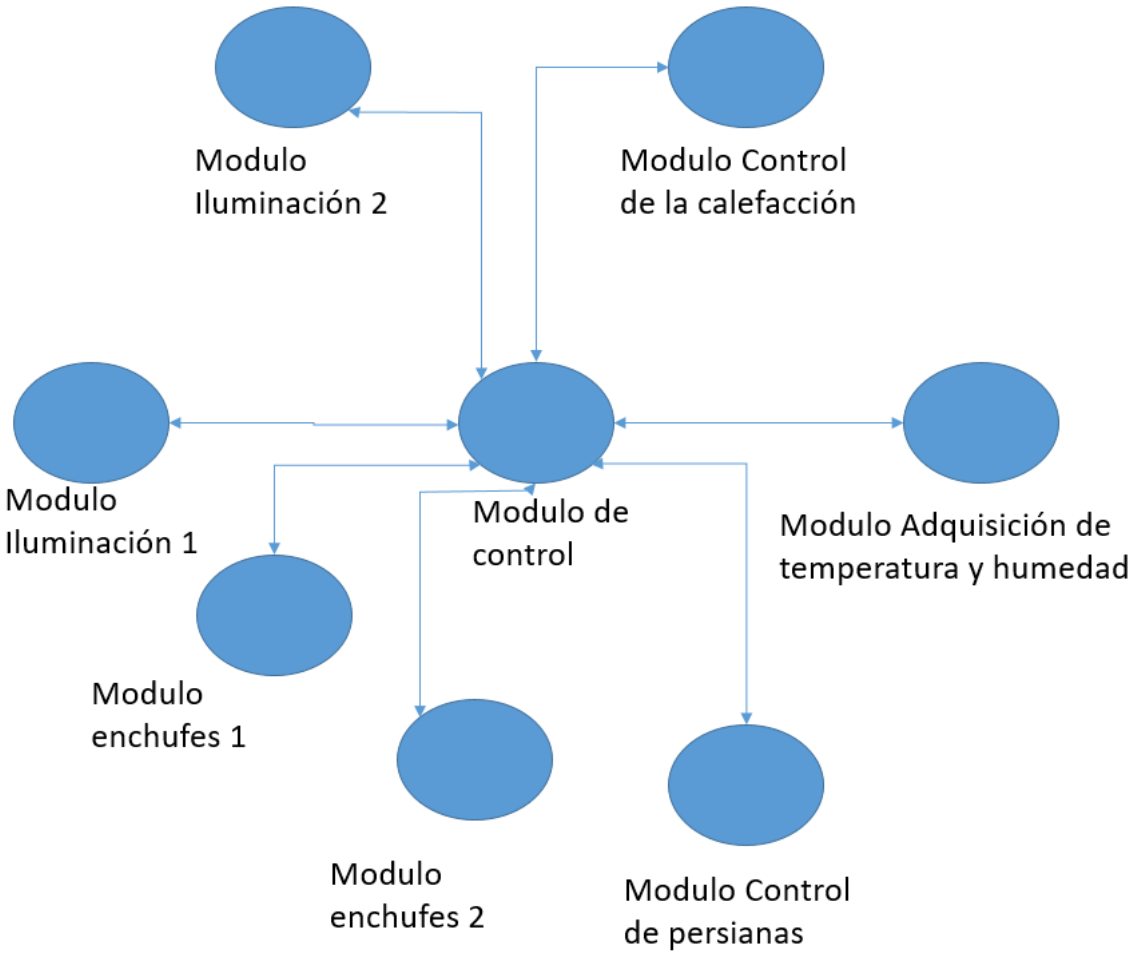

Figura 53. Esquema de conexión de los módulos alternativo

Los módulos de Persianas, Calefacción, Apertura de puerta, y Temperatura y humedad están pensados para que solo tengamos uno de ellos conectado al mismo tiempo a nuestro módulo de control, debido a las limitaciones de dispositivos del chip ESP8266 el sistema está pensado para un módulo de control por habitación y las habitaciones medias no tienen más de una ventana, y no es necesario más de un punto de toma de temperatura; y los de calefacción y apertura de puertas no son necesarios más de uno por vivienda dado a que solo hay una caldera en la vivienda y solo se controla la puerta principal de la vivienda.

El sistema está pensado para que puedan convivir distintos módulos de control en la vivienda, aunque los distintos módulos de control no tendrán comunicación entre sí ni estarán conectados a la red wifi de la vivienda, como se observa en la *Figura 54*, debido a los problemas ya explicados anteriormente, que surgen al intentar comunicarnos con un módulo y con la red wifi al mismo tiempo el cual se puede extender a intentar comunicar dos módulos de control entre si y los distintos módulos conectados a ellos.

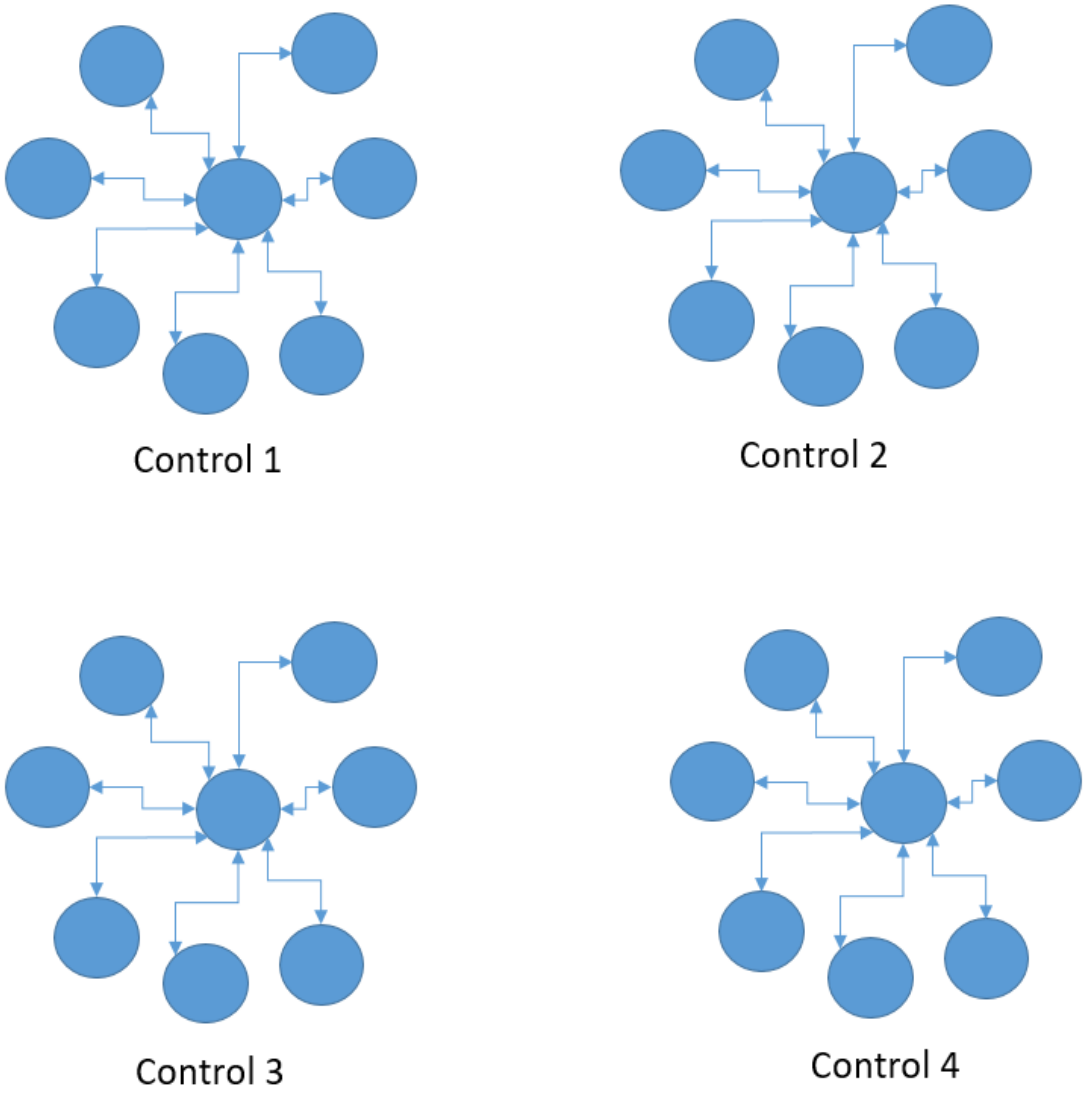

Figura 54. Esquema de conexión varios módulos de control

A continuación, se procederá a exponer los resultados obtenidos en con los distintos módulos de manera más extensa.

6.2 Módulos

En este apartado se explicarán los resultados módulo a módulo centrándose en las peculiaridades de cada uno, así como mostrando las páginas web específicas que genera cada uno de los módulos. Estas páginas web son completamente compatibles con ordenadores y teléfonos móviles por lo cual se podrá usar cualquiera de ellos para el control del sistema sin problemas de incompatibilidad por parte de las páginas web generadas por los módulos.

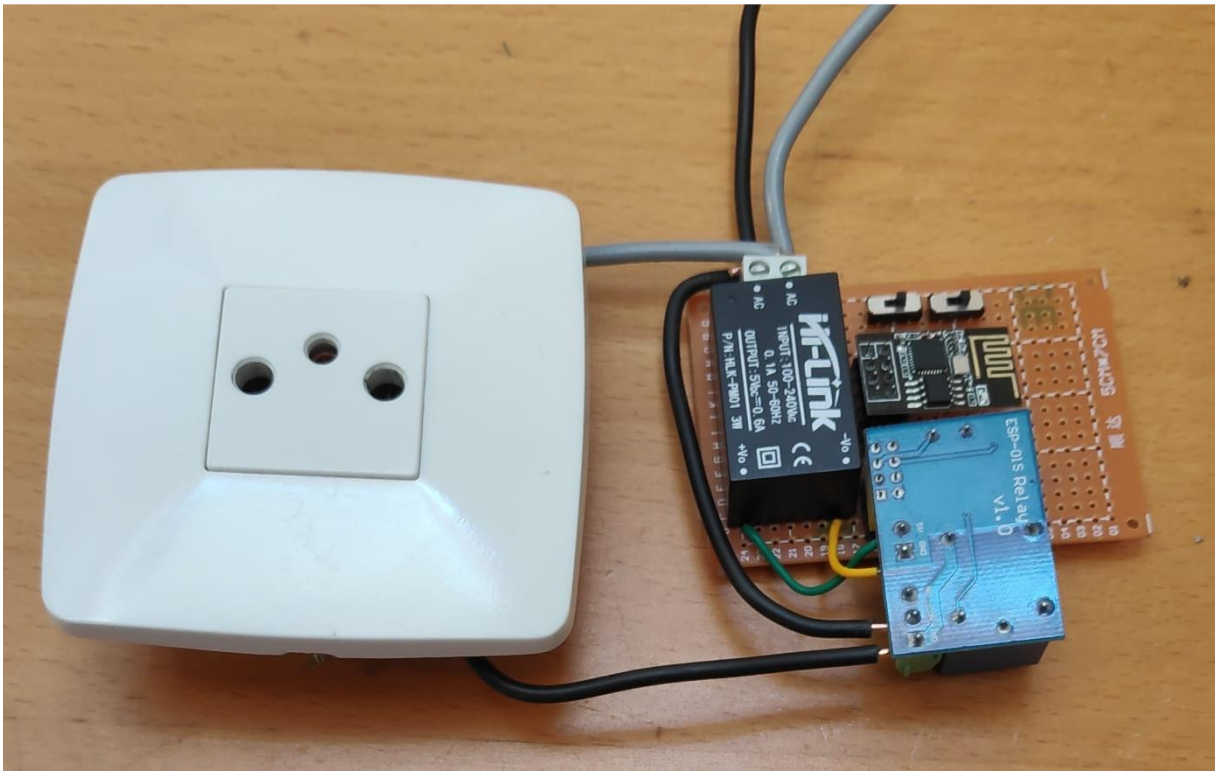

Figura 55. Módulo de control de enchufes conectado

.

Los ejemplos de páginas web mostrados en este capítulo son los obtenidos durante diversas pruebas de los módulos.

6.2.1 Iluminación

Este módulo se centra en el control de las luces. Su conexionado a la red eléctrica se puede observar en la *Figura 15.* En este conexionado se establece como se conecta el módulo para alimentarlo y su actuación sobre la luz o las luces pertinentes, este módulo

solo realiza la computación del relé de la posición de normalmente abierto a la posición de normalmente cerrado, para ello recibe la petición "/*gpio*".

La configuración del módulo consiste en:

Cambiar el nombre del módulo. Lo cual conseguimos a través del módulo de control el cual nos permitirá cambiarle el nombre.

Cambiar la red a la que se conecta el módulo. Para ello es necesario que esté conectado ya a una red, a continuación accederemos al módulo a través de la dirección IP local del módulo y añadiéndole la petición "/*conf*" lo cual nos generara la página web que se observa en la *Figura 56* la cual nos pide la introducción del nuevo SSID y de la nueva contraseña y le daremos al botón guardar para cambiarlos en el módulo.

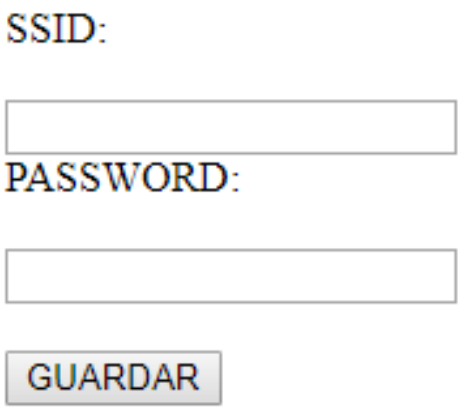

Figura 56. Página web, función "conf"

Debemos tener en cuenta que con la acción guardar el módulo se conectara directamente a la nueva red introducida, siempre que dicha red exista con los parámetros introducidos, dejando de tener acceso al módulo desde la red anterior, si la red introducida no existe el módulo se mantendrá en la red actual y no cambiara de red.

Esta es toda la configuración permitida por el módulo de iluminación.

6.2.2 Enchufes

Como ya se comentó en los capítulos anteriores el módulo de Iluminación y el módulo de Control de los enchufes son iguales en diseño, únicamente varia su conexión a la red eléctrica. No se realizará por tanto una nueva explicación dado que sería la misma a la comentada en el apartado *6.2.1 Iluminación.*

6.2.3 Persianas

Este módulo está diseñado para el control de la subida y bajada de las persianas. No obstante el diseño servirá para controlar cualquier sistema que conste de un motor que gire en dos direcciones y de dos sensores final de carrera como podría ser una puerta de garaje.

El conexionado de este módulo a la red lo podemos observar en la *Figura 18* como ya se ha comentado este módulo se conectará al motor de tal manera que la activación de un relé inicie el motor en giro horario y la activación del relé contrario active el motor en giro anti-horario.

Para estas activaciones el módulo deberá recibir las peticiones correspondientes que son "/*subir*" y "/*bajar*", el módulo también permite el control de la parada del motor mediante la petición "/*parar*".

Por seguridad el módulo tiene una parada automática en cuanto detecta que alguno de los dos sensores final de carrera se activa para no forzar el motor.

La configuración de este módulo es la misma a la del módulo de Iluminación, es decir permite cambiarle el nombre al módulo desde el módulo de control y permite cambiar la red a la cual se conectará el módulo como se explicó en el módulo de Iluminación.

6.2.4 Control de calefacción

Este módulo se centra en el control del encendido y apagado de la caldera, es decir manda la orden a la caldera para que esta caliente el agua de los radiadores o deje de calentarla, no controla el encendido del quemador, dado que eso podría resultar peligroso y de ello ya se encarga el controlador de la caldera. Solamente se mandará la misma señal que le mandaría un termostato para calentar los elementos de la calefacción.

El conexionado de este módulo para su funcionamiento esta descrito en la *Figura 19*.

La configuración que nos permite este módulo es la misma a la del módulo de Iluminación, es decir nos permite cambiarle el nombre al módulo y cambiarle la red a la cual se conecta.

6.2.5 Control de apertura de puerta

Este módulo está diseñado para actuar de forma paralela a un telefonillo, aunque puede funcionar junto con cualquier sistema de apertura electrónico de puertas.

Las conexiones de este módulo están detalladas en la *Figura 20.*

La configuración permitida por el módulo es la misma a la de los módulos anteriores, es decir, nos permite cambiarle el nombre al módulo y cambiarle la red a la cual se conectara, pero también nos permitirá modificar el tiempo que mantiene la apertura de la puerta, para ello genera la página web de la *Figura 57*, y como vemos la página web se parece mucho a la de cambio de la red pero en esta ocasión nos pide la introducción del tiempo que queremos y una vez le demos a guardar este tiempo se cambiara en el módulo.

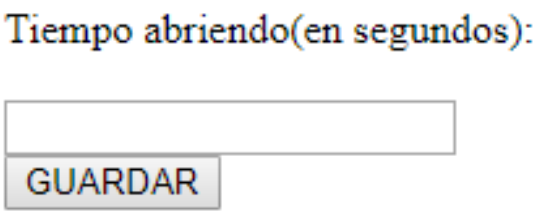

Figura 57. Página web cambiar tiempo

6.2.6 Adquisición de temperatura y humedad

Este módulo es el que más se diferencia del resto debido a que no actúa, sino que nos proporciona información de temperatura y humedad, aunque este módulo pueda funcionar sin necesidad de un módulo de control, no es aconsejable debido a que la manera en la que se visualizan los datos no es ni la más estética ni la más cómoda dado que la página web para mostrar los datos se genera en módulo de control, la salida directamente si accedemos a este módulo es la mostrada en la *Figura 58.*

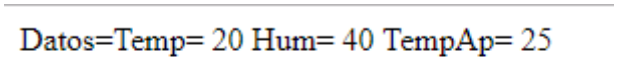

Figura 58. Página web módulo adquisición temperatura y humedad

La mayoría de los navegadores son capaces de leer los datos codificados y de mostrar la información, pero es posible que algún navegador no sea capaz y no se pueda ver en él esta información.

La conexión de este módulo esta detallada en la *Figura 21*, aunque esta conexión es interna dado que este módulo al no actuar no necesita más que la alimentación para funcionar correctamente.

Las posibilidades de configuración son las mismas de los módulos anteriormente comentados; nos permite cambiarle el nombre al módulo y cambiarle a la red a la cual se conectará.

6.2.7 Control

Este módulo, como ya se ha comentado anteriormente es completamente distinto a los módulos anteriores. Este módulo es el que coordina toda la red, aunque como se ha comentado no es imprescindible para el funcionamiento si es muy recomendable su inclusión.

La conexión de este módulo se realiza mediante conexión a la alimentación eléctrica.

Este módulo nos devuelve la página web principal cuando le mandamos la petición "/*general*". Esta página web tendrá el aspecto de la *Figura 59*, pero si ya hemos realizado la identificación de los dispositivos conectados se parecerá a la *Figura 60*, aunque su aspecto puede variar en función del número de dispositivos y el orden en el que se han conectado.

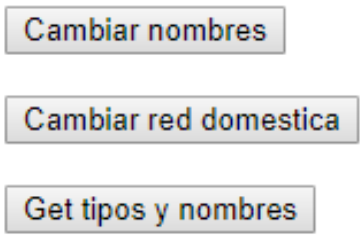

Figura 59. Página web principal vacía

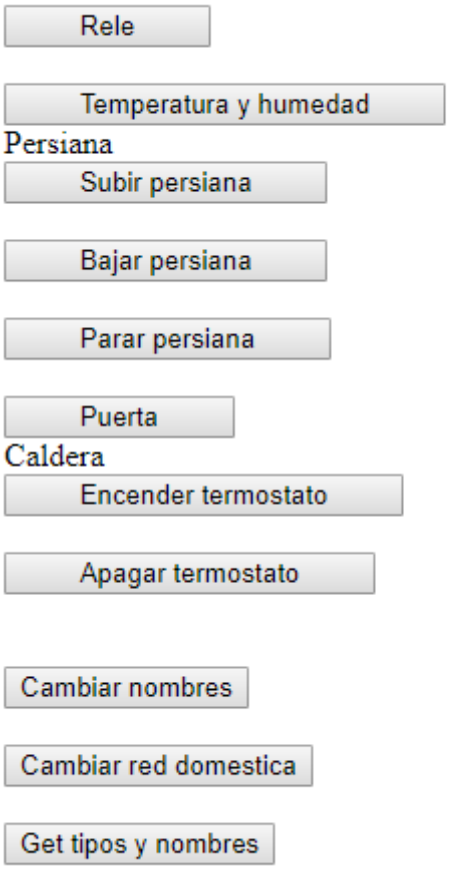

Figura 60. Página web principal llena

Desde esta página web principal se nos permitirá controlar todos los módulos conectados al módulo de control, los módulos de Apertura de puerta, Iluminación, Enchufes, y Adquisición de temperatura y humedad solamente nos presentaran un botón para su control, en cambio el módulo de Persianas nos presentara tres botones y el de Control de la calefacción dos.

A parte de los botones para el control de los módulos la página web principal nos mostrara los botones necesarios para acceder a la configuración y poder cambiar los nombres de los módulos, cambiar los parámetros de la red que genera el módulo de control y realizar la identificación de los dispositivos conectador.

Los únicos botones que generan una página web diferente a la principal son los del módulo de Adquisición de temperatura y humedad, y los botones de Cambiar nombres y Cambiar red doméstica. El resto de botones realizarán las operaciones correspondientes y se volverán a mostrar en la página web principal.

La página web mostrada por el botón de control del módulo de Adquisición de temperatura y humedad la se puede ver en la *Figura 61*, en ella se nos mostrara la temperatura y la humedad adquiridas por el módulo y nos mostrara otro botón para regresar a la página web principal.

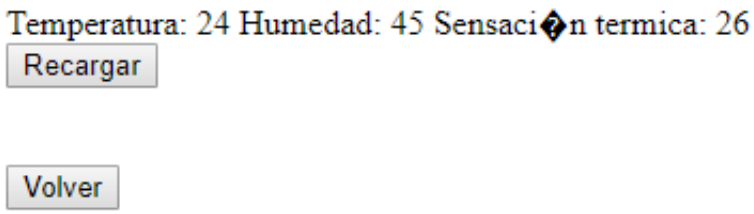

Figura 61. Página web temperatura y humedad

El botón de cambiar nombres nos mostrara una página web con el nombre actual de todos los dispositivos conectados y un hueco debajo para introducir el nuevo nombre, los módulos en los que se deje el espacio en blanco no cambiaran de nombre, también nos presenta un botón guardar el cual nos redirigirá a la página web principal guardando los cambios realizados mandándoselos al módulo mediante una petición "/*guardar_nombres*" y recogiéndolos mediante la función *"arg"* del servidor, como se puede ver en la *Figura 62*.

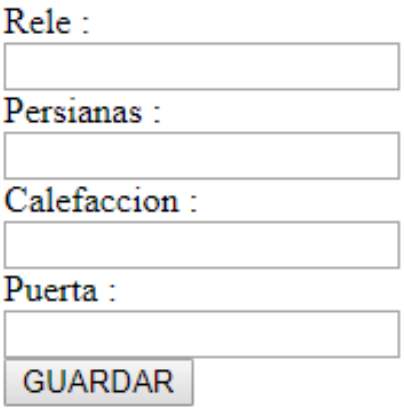

Figura 62. Página web cambiar nombre

El botón de cambiar red doméstica nos mostrara una página web donde se nos pedirán los datos de la nueva red como se observa en la *Figura 63*, así como un botón guardar para guardar la nueva configuración mandado los datos una petición al módulo "/*guardar_conf*" y recogiéndolos con la función *"arg"* del servidor y que nos redirija a la página web principal.

Figura 63. Página web cambiar red doméstica

7. PRESUPUESTO

7.1. Introducción

Uno de los principales objetivos de este proyecto es encontrar una solución de bajo costo. A continuación se detallan los costes que tendría una instalación de los distintos módulos.

7.2. Tarjetas

Las tarjetas de este proyecto han sido seleccionadas tanto por su funcionalidad y prestaciones como por su precio, por ello tampoco se utilizan en todos los módulos tarjetas NodeMCU, aunque estas sean las más cómodas de programar y alimentar, pero debido a su precio ligeramente superior a las tarjetas ESP-01s en aquellos módulos en los que es posible se han utilizado las tarjetas ESP-01s.

El coste de adquisición de las tarjetas NodeMCU oscila entre los 2€ y los 5€ dependiendo del proveedor y de si lleva los racks de pines soldados o no.

El coste de la placa ESP-01s oscila entre los 1,20€ los 3,50€ de media dependiendo de los proveedores.

El relé para el ESP-01s tiene un precio entre 0,80€ y 3,00€ en función del proveedor.

Existen packs en los cuales ya vienen juntos la tarjeta ESP-01s y el relé cuyo coste varía entre el 1,90€ y los 3,50€ dependiendo del proveedor.

El transformador Hi-Link tiene un coste entre el 1,90€ y los 3,80€ en función del proveedor elegido.

El sensor DHT22 tiene un precio que varía entre los 2,60€ y los 5,00€ dependiendo del proveedor que se elija.

Los transformadores para alimentar las tarjetas NodeMCU tienen un precio que oscila entre el 1,50€ y lo 3,00€ en función de los proveedores escogidos.

Los relés 5v tienen un coste de entre los 0.80€ y los 3,00€ en función del proveedor escogido.

Estas cifras son únicamente del coste de adquisición de las tarjetas y los periféricos. A esos precios hay que sumarle el coste de mano de obra y desarrollo, así como el coste de los materiales necesarios para una correcta instalación.

7.3 Presupuesto de implantación

A continuación, procederé a explicar cómo quedaría el presupuesto final de la instalación de un conjunto compuesto por un módulo de cada tipo en una vivienda.

Unidad	Descripción	Medición	Precio unitario	Precio final
	Capítulo 1: módulos			
	Módulo de control			
Ud.	Suministro del módulo de control, el cual consta de adaptador de corriente de micro USB y la tarjeta NodeMCU, dejándolo totalmente colocado y funcionando.	1	20,00€	20,00€
	Módulo relé			
Ud.	Suministro del módulo relé que se utiliza tanto en el control de las luces como en el control de los enchufes el cual comprende la tarjeta ESP-01s, la palca relé y el transformador HI-link, dejándolo totalmente montado y funcionando.	2	22,00€	44,00€
	Módulo de Apertura de puerta			
Ud.	Suministro del módulo de control de apertura de puerta el cual comprende la tarjeta ESP-01s, la palca relé y el transformador HI-link dejándolo completamente instalado y funcionando.	1	22,00€	22,00€

Unidad	Descripción	Medición	Precio unitario	Precio final
	Módulo de Control de calefacción			
Ud.	Suministro del módulo de Control de la calefacción el cual comprende la tarjeta ESP-01s, la palca relé y el transformador HI-link dejándolo completamente montado y funcionando.	1	22,00€	22,00€
	Módulo de Persianas			
Ud.	Suministro del módulo de control de las persianas el cual comprende la tarjeta NodeMCU el adaptador de corriente a microUSB y los relés necesarios en el montaje dejándolo completamente montado y funcionando.	1	30,00€	30,00€
	Módulo de Adquisición de temperatura y humedad			
Ud.	Suministro del módulo de adquisición de temperatura y humedad el cual comprende la tarjeta NodeMCU el adaptador de corriente a microUSB y el sensor DHT22, dejándolo completamente colocado y funcionando.	1	30,00€	30,00€
	Total del capitulo.....			168€

Unidad	Descripción	Medición	Precio unitario	Precio final
	Capítulo 2: Instalación			
	Mano de obra.			
h	Esto comprende toda la mano de obra necesaria para montar todos los módulos dejándolos funcionando.	4	60,00€	240,00€
	Otros			
	Aquí están comprendidos todos los gastos no fijos derivados de la instalación como pueden ser secciones de cable o clemas.		30,00€	30,00€
	Total del capitulo.....			270,00€
	Total del presupuesto (IVA no incluido).....			438,00€

En el presupuesto se ha supuesto el coste del módulo ya programado, aunque dado que se trata de código libre cualquier persona podría comprar los módulos y programarlos por su cuenta con lo que el coste sería mucho menor.

Como se observa en el presupuesto la instalación del sistema no llevará más de una mañana de mano de obra, así como no será necesaria la realización de ninguna obra en la vivienda, siendo esta únicamente opcional por temas estéticos.

Como se observa también en el presupuesto la mayor inversión a realizar sería la de la mano de obra, pero debido a la sencillez de instalación cualquier persona con conocimientos básicos de las instalaciones eléctricas de una vivienda podría realizar la instalación.

El presupuesto se podría reducir a únicamente el coste de los módulos por lo cual quedaría de la siguiente manera.

Unidad	Descripción	Medición	Precio unitario	Precio final
	Capítulo 1: módulos			
	Módulo de control			
Ud.	Suministro del módulo de control, el cual consta de adaptador de corriente de micro USB y la tarjeta NodeMCU, dejándolo totalmente colocado y funcionando.	1	20,00€	20,00€
	Módulo relé			
Ud.	Suministro del módulo relé que se utiliza tanto en el control de las luces como en el control de los enchufes el cual comprende la tarjeta ESP-01s, la palca relé y el transformador HI-link, dejándolo totalmente montado y funcionando.	2	22,00€	44,00€
	Módulo de Apertura de puerta			
Ud.	Suministro del módulo de control de apertura de puerta el cual comprende la tarjeta ESP-01s, la palca relé y el transformador HI-link dejándolo completamente instalado y funcionando.	1	22,00€	22,00€

Unidad	Descripción	Medición	Precio unitario	Precio final
	Módulo de Control de calefacción			
Ud.	Suministro del módulo de Control de la calefacción el cual comprende la tarjeta ESP-01s, la palca relé y el transformador HI-link dejándolo completamente montado y funcionando.	1	22,00€	22,00€
	Módulo de Persianas			
Ud.	Suministro del módulo de control de las persianas el cual comprende la tarjeta NodeMCU el adaptador de corriente a microUSB y los relés necesarios en el montaje dejándolo completamente montado y funcionando.	1	30,00€	30,00€
	Módulo de Adquisición de temperatura y humedad			
Ud.	Suministro del módulo de adquisición de temperatura y humedad el cual comprende la tarjeta NodeMCU el adaptador de corriente a microUSB y el sensor DHT22, dejándolo completamente colocado y funcionando.	1	30,00€	30,00€
	Total del capitulo.....			168,00€
	Total del presupuesto (IVA no incluido).....			168,00€

En este presupuesto se he utilizado la media aproximada del coste de los dispositivos mencionados en el apartado *7.2. Tarjetas* Por lo cual es probable que según los proveedores que se consulten pueda salir un resultado diferente ya sea mayor o menor.

En estos presupuestos se ha calculado un coste aproximado de desarrollo y de mano de obra de un 100% de coste de los componentes lo cual nos deja un 30% de beneficio sobre el pecio final.

El coste de mano de obra anteriormente mencionado es únicamente el del montaje de los módulos con los componentes adecuados y su programación

7.4 Presupuesto de fabricación

A continuación, se detallarán los constes de fabricación de los módulos, en este presupuesto se realizará de igual manera que los anteriores en el cual las medidas serán las necesarias para fabricar un módulo de cada tipo.

Unidad	*Descripción*	*Medición*	*Precio unitario*	*Precio total*
	Capítulo 1: Costes fijos			
	Ordenador:			
Ud.	Ordenador con puertos USB y el arduino IDE instalado, para programar los módulos.	1	800,00€	800,00€
	Soldador de estaño			
Ud.	Soldador de estaño para soldar los componentes de los módulos.	1	50,00€	50,00€
	Coste de desarrollo			
Mes	Un ingeniero trabajando para desarrollar los módulos.	3	2.400,00€	7.200,00€

Unidad	Descripción	Medición	Precio unitario	Precio total
	Costes indirectos			
	Uso de instalaciones y mantenimiento de las mismas junto con los gastos de oficina.		1.800,00€	1.800,00€
	Total del capitulo.....			**9.850,00€**
	Capítulo 2: Materiales			
Ud.	**Tarjeta NodeMCU**	3	4,00€	12,00€
Ud.	**Tarjeta ESP-01s**	4	2,00€	8,00€
Ud.	**Placa relé ESP-01s**	4	1,50€	6,00€
Ud.	**Relé 5v**	2	1,30€	2,60€
Ud.	**Transformador Hi-Link**	4	2,50€	10,00€
Ud.	**Sensor DHT22**	1	3,50€	3,50€
Ud.	**Transformador de corriente**	3	2,00€	6,00€
	Otros gastos			
	Los cuales comprenden el gasto en cableado, interruptores, racks de pines, estaño, etc.		10,00€	10,00€
	Total del capitulo.....			**58,10€**

Unidad	Descripción	Medición	Precio unitario	Precio total
	Capítulo 3: Montaje			
	Coste de montaje			
Módulo	Coste del personal necesario, durante el tiempo necesario para montar un módulo.	7	7,00€	49,00€
	Total del capitulo.....			**49,00€**
	Total del presupuesto (IVA no incluido).....			**9.957,10€**

Como se observa en este presupuesto el coste de fabricación de estos siete módulos quedaría bastante elevado y no coincidiría con el coste por modulo expuesto en el apartado anterior.

Esto se debe a los costes fijos del proyecto, pero si en lugar de fabricar únicamente esos 7 módulos se fabricasen 10.000 módulos el coste se reduciría considerablemente al expuesto en el apartado anterior otorgando ya un margen de beneficio del 30% aproximadamente, si la producción se realizase en una planta industrial mediante un proceso automatizado y utilizando tarjetas impresas en vez de realizar la conexión mediante cables convencionales el coste se reduciría mucho más aumentando los márgenes de beneficio significativamente.

8. CONCLUSIONES Y TRABAJOS FUTUROS

8.1 Conclusiones

Los objetivos de este proyecto eran conseguir desarrollar un sistema domótico que tuviera un precio inferior por módulo al de los sistemas domóticos actuales y que no necesitara de una obra o de canaletas para su instalación.

En este aspecto se han conseguido los objetivos dado a que al usar la tecnología wifi para la comunicación de los módulos no es necesario para que los módulos se comuniquen realizar una nueva canalización en la pared u ocultar los cables mediante canaletas.

Con respecto al otro objetivo principal de que el coste de esta domotización sea más barato que la de los sistemas actuales también se ha cumplido. Como se ha visto en el presupuesto el precio de instalar un módulo de cada de los que se han diseñado ronda los 438.00€ mientras que los sistemas cableados como pueden ser los sistemas KNX [7] o Lonworks [4] tienen un coste por dispositivo de entre los 100€ y los 300€ por lo cual el sistema desarrollado es mucho más barato que los sistemas actualmente en el mercado. Los sistemas más parecidos al desarrollado en este proyecto serían los sistemas Zigbee [8] los cuales son sistemas inalámbricos, que tienen un coste de entre 40€ y 300€ cada módulo, por lo que el sistema desarrollado en este proyecto sigue siendo más barato, además a los precios descritos anteriormente habría que sumarle el coste de instalación, este coste suele representar entre un 20% y un 50% dependiendo de la complejidad de la instalación y de si será necesario realizar obra para la instalación.

8.2. Líneas futuras de trabajo

Las posibles líneas de ampliación del proyecto serian por un lado conseguir un módulo que permitiese la conexión de un número superior de dispositivos para poder controlar más sistemas en la casa, también sería conveniente conseguir un módulo que se pudiera conectar a la red wifi de la vivienda sin que saltara el Watchdog de la placa al esperar la respuesta de las otras tarjetas.

Debido a las propias funciones del chip ESP8266 la conexión y el envío de las peticiones desde el módulo de control y los módulos conectados a el tiempo de respuesta del sistema ronda los 8 segundos los cual no es muy cómodo a la hora de controlar los sistemas de una vivienda por lo cual una de las líneas de trabajo futuro podría ser la de optimizar las librerías del ESP8266 para reducir el tiempo que se tarda en enviar la petición y con ello reducir el tiempo de respuesta del sistema.

Se podría considerar buscar la manera de reducir el tamaño de los módulos de tal manera que se pudiesen integrar en los mismos agujeros de la pared dedicados para los sistemas tradicionales pudiendo convivir con ellos en el mismo espacio.

Otra de las posibles líneas de trabajo podría ser realizar un rediseño de las páginas web mostradas para obtener un resultado mucho más estético y agradable, mediante la programación con CSS.

Estas serían las principales líneas de trabajo futuro que se han observado, aunque no serían las únicas, con la aparición de nuevos materiales y tecnologías este proyecto puede desarrollar otras líneas de trabajo futuro a medida que va implementándose.

REFERENCIAS

[1] REAL ACADEMIA ESPAÑOLA, «Diccionario de la lengua española,» [En línea]. Available: https://dle.rae.es. [Último acceso: 22 Mayo 2019].

[2] Asociacion Española de Domótica e Inmótica, «CEDOM:Sobre Dómotica: Que es Dómotica,» [En línea]. Available: http://www.cedom.es/sobre-domotica/que-es-domotica. [Último acceso: 02 Mayo 2019].

[3] Wikipedia, «X10,» [En línea]. Available: https://es.wikipedia.org/wiki/X10. [Último acceso: 06 Junio 2019].

[4] LonMark, «LonMark cono sur,» [En línea]. Available: https://www.lonmark.la/plataforma-lonworks/. [Último acceso: 6 Junio 2019].

[5] American Society of Heating, Refrigerating and Air-Conditioning Engineers, «BACnet,» [En línea]. Available: http://www.bacnet.org/. [Último acceso: 06 Junio 2019].

[6] Asocioacion española PROFIBUS / PROFINET / IOLINK, «Profibus,» [En línea]. Available: http://profibus.es/profibus. [Último acceso: 06 Junio 2019].

[7] Asociacion KNX, «KNX,» [En línea]. Available: https://www.knx.org/es/. [Último acceso: 06 Junio 2019].

[8] Zigbee Alliance, «Zigbee,» [En línea]. Available: https://www.zigbee.org/. [Último acceso: 06 Junio 2019].

[9] EnOcean, «EnOcean,» [En línea]. Available: https://www.enocean.com/. [Último acceso: 06 Junio 2019].

[10] Ministerio de Industria, Comercio y Turismo, «Reglamento Electrotécnico de Baja Tensión,» [En línea]. Available: http://www.f2i2.net/legislacionseguridadindustrial/Si_Ambito.aspx?id_am=76. [Último acceso: 22 Mayo 2019].

[11] Asociación Española de Normalización y Certificación, «Reglamento Particular de la marca AENOR para instalaciones de sistemas Domóticos en Viviendas,» 2007.

[12] Asociación Española de Normalización y Certificación, «EA0026:2006 Prescripciones generales de instalación y evaluación,» 2006.

[13] Boletin Oicial del Estado, «Documento Básico DB-HE "Ahorro de Energía", del codigo tecnico de edificación,» [En línea]. Available: http://www.boe.es/boe/dias/2013/09/12/pdfs/BOE-A-2013-9511.pdf. [Último acceso: 22 Mayo 2019].

[14] Asociacion española de domotica e inmotica, «CEDOM,» [En línea]. Available: http://www.cedom.es/media/k2/items/cache/d857df8a9cfd7c86ccb2e9f710b5414b_XL.jpg. [Último acceso: 06 Junio 2019].

[15] Iberdrola, «Iberdrola,» [En línea]. Available: https://www.iberdrola.com/te-interesa/tecnologia/domotica. [Último acceso: 06 Junio 2019].

[16] L. Llamas, «Luis Llamas,» [En línea]. Available: https://www.luisllamas.es/. [Último acceso: 06 Junio 2019].

[17] *Apuntes de la asignatura Aplicaciones de la automatizacion en edificios.*

ANEXOS

Anexo código

Código completo Módulo Iluminación y Enchufes

```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>

String ssid = "Domotica";
String password="";
String tipodisp="1";//0=Desconocido,1=rele,2=dht,3=persianas,4=telefonillo,5=termostato,6=Control
String nombre="Rele";
int Pin_rele=0;
bool estado=true;

ESP8266WebServer server(80);
void conecxion(){
}
void sendtipo(){ // Funcion para mandar al control el tipo de placa que es
  String aux="";
  aux+="Datos=Tipo=";
  aux+=tipodisp;
  server.send(200,"text/html",aux);
}
void rele(){  // Funcion para controlar el rele
  !estado; // Invierte el estado del rele
  if (estado){
    digitalWrite(Pin_rele,HIGH);
  }else{
    digitalWrite(Pin_rele,LOW);
  }
  Serial.println("Rele");
}
void setnombre(){ // Funcion para establecer el nuevo nombre de la plca
  Serial.println("Set nombre");
  nombre=server.arg("nombre");
}
void getnombre(){ // Funcion que manda el nombre de la placa al control
  Serial.println("Get nombre");
  String aux;
  aux="Datos=Nombre=";
  aux+=nombre;
  server.send(200,"text/html",aux);
}
```

```
void conf(){  // Funcion que genera la pagina web para cambiar la red a la que se conecta
  Serial.println("config");
  server.send(200,"text/html","<!DOCTYPE html>"
                              "<html>"
                              "<head>"
                              "<title>Configuración</title>"
                              "<meta charset='UTF-8'>"
                              "</head>"
                              "<body>"
                              "</form>"
                              "<form action='guardar_conf' method='get'>"
                              "SSID:<br><br>"
                              "<input class='inputl' name='ssid' type='text'><br>"
                              "PASSWORD:<br><br>"
                              "<input class='inputl' name='pass' type='text'><br><br>"
                              "<input class='boton' type='submit' value='GUARDAR'/><br><br>"
                              "</form>"
                              "</body>"
                              "</html>");
}
void guardar(){ // Funcion para guardar la nueva red y conectarse
  ssid=server.arg("ssid");
  password=server.arg("pass");
  if(ssid==""){
    Serial.println("Sin red");
  }else{
    if (password==""){
      if (WiFi.begin(ssid)){
        Serial.println("Conectado");
        Serial.println(WiFi.localIP());
      }else{
        Serial.println("Fallo de conexion");
      }
    }else{
      if (WiFi.begin(ssid,password)){
        Serial.println("Conectado");
        Serial.println(WiFi.localIP());
      }else{
        Serial.println("Fallo de conexion");
      }
    }
  }
}
```

```
83  void setup() {
84    Serial.begin(115200);
85    WiFi.mode(WIFI_STA);
86    if(WiFi.begin(ssid)){
87      Serial.println("conectado");
88    }else{
89      Serial.println("Fallo de conecxion");
90    }
91    server.on("/",conecxion);
92    server.on("/tipo",sendtipo);
93    server.on("/gpio",rele);
94    server.on("/setnombre",setnombre);
95    server.on("/getnombre",getnombre);
96    server.on("/conf",conf);
97    server.on("/guardar_conf",guardar);
98    server.begin();
99    pinMode(Pin_rele,OUTPUT);
100 }
101
102 void loop() {
103   server.handleClient();
104 }
```

Código completo Módulo Persianas

```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>

String ssid = "Domotica";
String password="";
String tipodisp="3";//0=Desconocido,1=rele,2=dht,3=persianas,4=telefonillo,5=termostato,6=Control
String nombre="Persiana";
const int pinSubir=5;
const int pinBajar=4;
const int pinSensorArriba=0;
const int pinSensorAbajo=2;

ESP8266WebServer server(80);

void conecxion(){
}
void sendtipo(){ // Funcion para mandar al control el tipo de placa que es
  String aux="";
  aux+="Datos=Tipo=";
  aux+=tipodisp;
  server.send(200,"text/html",aux);
}
void setnombre(){ // Funcion para establecer el nuevo nombre de la plca
  Serial.println("Set nombre");
  nombre=server.arg("nombre");
}
void getnombre(){ // Funcion que manda el nombre de la placa al control
  Serial.println("Get nombre");
  String aux;
  aux="Datos=Nombre=";
  aux+=nombre;
  server.send(200,"text/html",aux);
}
```

```
void conf(){ // Funcion que genera la pagina web para cambiar la red a la que se conecta
  Serial.println("config");
  server.send(200,"text/html","<!DOCTYPE html>"
                              "<html>"
                              "<head>"
                              "<title>Configuración</title>"
                              "<meta charset='UTF-8'>"
                              "</head>"
                              "<body>"
                              "</form>"
                              "<form action='guardar_conf' method='get'>"
                              "SSID:<br><br>"
                              "<input class='input1' name='ssid' type='text'><br>"
                              "PASSWORD:<br><br>"
                              "<input class='input1' name='pass' type='text'><br><br>"
                              "<input class='boton' type='submit' value='GUARDAR'/><br><br>"
                              "</form>"
                              "</body>"
                              "</html>");
}
void guardar(){ // Funcion para guardar la nueva red y conectarse
  ssid=server.arg("ssid");
  password=server.arg("pass");
  if(ssid==""){
    Serial.println("Sin red");
  }else{
    if (password==""){
      if (WiFi.begin(ssid)){
        Serial.println("Conectado");
        Serial.println(WiFi.localIP());
      }else{
        Serial.println("Fallo de conexion");
      }
    }else{
      if (WiFi.begin(ssid,password)){
        Serial.println("Conectado");
        Serial.println(WiFi.localIP());
      }else{
        Serial.println("Fallo de conexion");
      }
    }
  }
}
```

```
77  void subir(){ // Funcion para subir la persiana
78    Serial.println("Subir");
79    digitalWrite(pinSubir,HIGH);
80    digitalWrite(pinBajar,LOW);
81    delay(1000);
82  }
83  void bajar(){ // Funcion para bajar la persiana
84    Serial.println("Bajar");
85    digitalWrite(pinSubir,LOW);
86    digitalWrite(pinBajar,HIGH);
87    delay(1000);
88  }
89  void parar(){ // Funcion para parar la persiana
90    Serial.println("Parar");
91    digitalWrite(pinSubir,LOW);
92    digitalWrite(pinBajar,LOW);
93  }
94
95  void setup() {
96    Serial.begin(115200);
97    WiFi.mode(WIFI_STA);
98    if(WiFi.begin(ssid)){
99      Serial.println("conectado");
100   }else{
101     Serial.println("Fallo de conecxion");
102   }
103   server.on("/",conecxion);
104   server.on("/tipo",sendtipo);
105   server.on("/setnombre",setnombre);
106   server.on("/getnombre",getnombre);
107   server.on("/conf",conf);
108   server.on("/guardar_conf",guardar);
109   server.on("/subir", subir);
110   server.on("/bajar", bajar);
111   server.on("/parar", parar);
112   server.begin();
113 }
114
115 void loop() {
116    server.handleClient();
117    boolean arriba=digitalRead(pinSensorArriba);
118    boolean abajo=digitalRead(pinSensorAbajo);
119    if(arriba==true||abajo==true){
120       digitalWrite(pinSubir,LOW);
121       digitalWrite(pinBajar,LOW);
122    }
123 }
```

Código completo Módulo Control de calefacción

```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>

String ssid = "Domotica";
String password="";
String tipodisp="5";//0=Desconocido,1=rele,2=dht,3=persianas,4=telefonillo,5=termostato,6=Control
String nombre="Termostato";
int Pin_rele=0;

ESP8266WebServer server(80);

void conecxion(){
}
void sendtipo(){ // Funcion para mandar al control el tipo de placa que es
  String aux="";
  aux+="Datos=Tipo=";
  aux+=tipodisp;
  server.send(200,"text/html",aux);
}
void setnombre(){ // Funcion para establecer el nuevo nombre de la palca
  Serial.println("Set nombre");
  nombre=server.arg("nombre");
}
void getnombre(){ // Funcion que manda el nombre de la placa al control
  Serial.println("Get nombre");
  String aux;
  aux="Datos=Nombre=";
  aux+=nombre;
  server.send(200,"text/html",aux);
}
```

```
void conf(){ // Funcion que genera la pagina web para cambiar la red a la que se conecta
  Serial.println("config");
  server.send(200,"text/html","<!DOCTYPE html>"
                              "<html>"
                              "<head>"
                              "<title>Configuración</title>"
                              "<meta charset='UTF-8'>"
                              "</head>"
                              "<body>"
                              "</form>"
                              "<form action='guardar_conf' method='get'>"
                              "SSID:<br><br>"
                              "<input class='inputl' name='ssid' type='text'><br>"
                              "PASSWORD:<br><br>"
                              "<input class='inputl' name='pass' type='text'><br><br>"
                              "<input class='boton' type='submit' value='GUARDAR'/><br><br>"
                              "</form>"
                              "</body>"
                              "</html>");
}
void guardar(){ // Funcion para guardar la nueva red y conectarse
  ssid=server.arg("ssid");
  password=server.arg("pass");
  if(ssid==""){
    Serial.println("Sin red");
  }else{
    if (password==""){
      if (WiFi.begin(ssid)){
        Serial.println("Conectado");
        Serial.println(WiFi.localIP());
      }else{
        Serial.println("Fallo de conexion");
      }
    }else{
      if (WiFi.begin(ssid,password)){
        Serial.println("Conectado");
        Serial.println(WiFi.localIP());
      }else{
        Serial.println("Fallo de conexion");
      }
    }
  }
}
void encender(){ // Funcion para encender el termostato
  digitalWrite(Pin_rele,HIGH);
  Serial.println("on");
}
```

```
78  void apagar(){ // Funcion para apagar el termostato
79    digitalWrite(Pin_rele,LOW);
80    Serial.println("off");
81  }
82  void setup() {
83     Serial.begin(115200);
84    WiFi.mode(WIFI_STA);
85    if(WiFi.begin(ssid)){
86      Serial.println("conectado");
87    }else{
88      Serial.println("Fallo de conecxion");
89    }
90    server.on("/",conecxion);
91    server.on("/tipo",sendtipo);
92    server.on("/setnombre",setnombre);
93    server.on("/getnombre",getnombre);
94    server.on("/conf",conf);
95    server.on("/guardar_conf",guardar);
96    server.on("/encender",encender);
97    server.on("/apagar",apagar);
98    server.begin();
99    pinMode(Pin_rele,OUTPUT);
100 }
101
102 void loop() {
103   server.handleClient();
104 }
```

Código completo Módulo Apertura de puerta

```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>

String ssid = "Domotica";
String password="";
String tipodisp="4";//0=Desconocido,1=rele,2=dht,3=persianas,4=telefonillo,5=termostato,6=Control
String nombre="Telefonillo";
int Pin_rele=0;
int tiempoabrir=1;

ESP8266WebServer server(80);

void conecxion(){
}
void sendtipo(){ // Funcion para mandar al control el tipo de placa que es
  String aux="";
  aux+="Datos=Tipo=";
  aux+=tipodisp;
  server.send(200,"text/html",aux);
}
void setnombre(){ // Funcion para establecer el nuevo nombre de la palca
  Serial.println("Set nombre");
  nombre=server.arg("nombre");
}
void getnombre(){ // Funcion que manda el nombre de la placa al control
  Serial.println("Get nombre");
  String aux;
  aux="Datos=Nombre=";
  aux+=nombre;
  server.send(200,"text/html",aux);
}
```

```
void conf(){ // Funcion que genera la pagina web para cambiar la red a la que se conecta
  Serial.println("config");
  server.send(200,"text/html","<!DOCTYPE html>"
                              "<html>"
                              "<head>"
                              "<title>Configuración</title>"
                              "<meta charset='UTF-8'>"
                              "</head>"
                              "<body>"
                              "</form>"
                              "<form action='guardar_conf' method='get'>"
                              "SSID:<br><br>"
                              "<input class='inputl' name='ssid' type='text'><br>"
                              "PASSWORD:<br><br>"
                              "<input class='inputl' name='pass' type='text'><br><br>"
                              "<input class='boton' type='submit' value='GUARDAR'/><br><br>"
                              "</form>"
                              "</body>"
                              "</html>");
}
void guardar(){ // Funcion para guardar la nueva red y conectarse
  ssid=server.arg("ssid");
  password=server.arg("pass");
  if(ssid==""){
    Serial.println("Sin red");
  }else{
    if (password==""){
      if (WiFi.begin(ssid)){
        Serial.println("Conectado");
        Serial.println(WiFi.localIP());
      }else{
        Serial.println("Fallo de conexion");
      }
    }else{
      if (WiFi.begin(ssid,password)){
        Serial.println("Conectado");
        Serial.println(WiFi.localIP());
      }else{
        Serial.println("Fallo de conexion");
      }
    }
  }
}
```

```
void abrir(){ // Funcion para abrir el telefonillo
  Serial.println("abrir");
  digitalWrite(Pin_rele,HIGH);
  delay((tiempoabrir*1000));
  digitalWrite(Pin_rele,LOW);
}
void tiempo(){ // Funcion que genera una pagina web para cambiar el tiempo que el telefonillo esta abriendo
  Serial.println("config tiempo");
  server.send(200,"text/html","<!DOCTYPE html>"
                              "<html>"
                              "<head>"
                              "<title>Configuración</title>"
                              "<meta charset='UTF-8'>"
                              "</head>"
                              "<body>"
                              "</form>"
                              "<form action='guardar_tiempo' method='get'>"
                              "Tiempo abriendo(en segundos):<br><br>"
                              "<input class='input1' name='tiempo' type='text'><br>"
                              "<input class='boton' type='submit' value='GUARDAR'/><br><br>"
                              "</form>"
                              "</body>"
                              "</html>");
}
void guardar_tiempo(){ // Funcion para guardar el nuevo tiempo de apertura
  tiempoabrir=(int)server.arg("tiempo").toInt();
}

void setup() {
  Serial.begin(115200);
  WiFi.mode(WIFI_STA);
  if(WiFi.begin(ssid)){
    Serial.println("conectado");
  }else{
    Serial.println("Fallo de conecxion");
  }
  server.on("/",conecxion);
  server.on("/tipo",sendtipo);
  server.on("/gpio", abrir);
  server.on("/setnombre",setnombre);
  server.on("/getnombre",getnombre);
  server.on("/conf",conf);
  server.on("/guardar_conf",guardar);
  server.on("/tiempo",tiempo);
  server.on("/guardar_tiempo",guardar_tiempo);
  server.begin();
}

void loop() {
  server.handleClient();
}
```

Código completo Módulo de Adquisición de temperatura y humedad

```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>

String ssid = "Domotica";
String password="";
String tipodisp="2";//0=Desconocido,1=rele,2=dht,3=persianas,4=telefonillo,5=termostato,6=Control
String nombre="DHT";
int Pin_rele=0;

ESP8266WebServer server(80);

#include "DHT.h"
#define DHTPIN 4
#define DHTTYPE DHT22

DHT dht(DHTPIN, DHTTYPE);
float h;
float t;

void conecxion(){
}
void sendtipo(){ // Funcion para mandar al control el tipo de placa que es
  String aux="";
  aux+="Datos=Tipo=";
  aux+=tipodisp;
  server.send(200,"text/html",aux);
}
void setnombre(){ // Funcion para establecer el nuevo nombre de la plca
  Serial.println("Set nombre");
  nombre=server.arg("nombre");
}
void getnombre(){ // Funcion que manda el nombre de la placa al control
  Serial.println("Get nombre");
  String aux;
  aux="Datos=Nombre=";
  aux+=nombre;
  server.send(200,"text/html",aux);
}
```

```
void conf(){ // Funcion que genera la pagina web para cambiar la red a la que se conecta
  Serial.println("config");
  server.send(200,"text/html","<!DOCTYPE html>"
                              "<html>"
                              "<head>"
                              "<title>Configuración</title>"
                              "<meta charset='UTF-8'>"
                              "</head>"
                              "<body>"
                              "</form>"
                              "<form action='guardar_conf' method='get'>"
                              "SSID:<br><br>"
                              "<input class='inputl' name='ssid' type='text'><br>"
                              "PASSWORD:<br><br>"
                              "<input class='inputl' name='pass' type='text'><br><br>"
                              "<input class='boton' type='submit' value='GUARDAR'/><br><br>"
                              "</form>"
                              "</body>"
                              "</html>");
}
void guardar(){ // Funcion para guardar la nueva red y conectarse
  ssid=server.arg("ssid");
  password=server.arg("pass");
  if(ssid==""){
    Serial.println("Sin red");
  }else{
    if (password==""){
      if (WiFi.begin(ssid)){
        Serial.println("Conectado");
        Serial.println(WiFi.localIP());
      }else{
        Serial.println("Fallo de conexion");
      }
    }else{
      if (WiFi.begin(ssid,password)){
        Serial.println("Conectado");
        Serial.println(WiFi.localIP());
      }else{
        Serial.println("Fallo de conexion");
      }
    }
  }
}
```

```
82 void datos(){ // Funcion para mandar los datos de temperatura y humedad al control
83   String aux="";
84   do{
85     h = dht.readHumidity();
86     t = dht.readTemperature();
87     if (isnan(h) || isnan(t)) {
88       Serial.println("Failed to read from DHT sensor!");
89     }
90   }while(isnan(h) || isnan(t));
91   float hic = dht.computeHeatIndex(t, h, false);
92   aux+="Datos=Temp=";
93   aux+=t;
94   aux+="Hum=";
95   aux+=h;
96   aux+="TempAp=";
97   aux+=hic;
98   server.send(200,"text/html",aux);
99 }
100
101 void setup() {
102   Serial.begin(115200);
103   WiFi.mode(WIFI_STA);
104   if(WiFi.begin(ssid)){
105     Serial.println("conectado");
106   }else{
107     Serial.println("Fallo de conecxion");
108   }
109   server.on("/",conecxion);
110   server.on("/tipo",sendtipo);
111   server.on("/dht",datos);
112   server.on("/setnombre",setnombre);
113   server.on("/getnombre",getnombre);
114   server.on("/conf",conf);
115   server.on("/guardar_conf",guardar);
116   server.begin();
117 }
118 void loop() {
119   server.handleClient();
120 }
```

Código completo Módulo Control

```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>

String ssid = "Domotica";
String ssid2="";
String password2="";
String ips[8]={"192.168.4.2","192.168.4.3","192.168.4.4","192.168.4.5","192.168.4.6","192.168.4.7","192.168.4.8","192.168.4.9"};
String nombres[8]={"Dispositivo 1","Dispositivo 2","Dispositivo 3","Dispositivo 4","Dispositivo 5","Dispositivo 6","Dispositivo 7","Dispositivo 8"};
String boton[8]={"data1=0","data2=0","data3=0","data4=0","data5=0","data6=0","data7=0","data8=0"};
String funcion[8]={"/data1=0","/data2=0","/data3=0","/data4=0","/data5=0","/data6=0","/data7=0","/data8=0"};
int tipo[8]={0,0,0,0,0,0,0,0};//0=Desconocido,1=rele,2=dht,3=persianas,4=telefonillo,5=termostato,6=Control
String tipodisp="6";
float humedad;
float temperatura;
float temperaturaAparente;

ESP8266WebServer server(80);

void conecxion(){
}
void general(){ // Funcion que manda la pagina web principal
  paginawebprincipal();
}
void conf(){  // Funcion para cambiar la red que genera
   server.send(200,"text/html","<!DOCTYPE html>"
                               "<html>"
                               "<head>"
                               "<title>Configuración</title>"
                               "<meta charset='UTF-8'>"
                               "</head>"
                               "<body>"
                               "</form>"
                               "<form action='guardar_conf' method='get'>"
                               "SSID:<br><br>"
                               "<input class='input1' name='ssid' type='text'><br>"
                               "<input class='boton' type='submit' value='GUARDAR'/><br><br>"
                               "</form>"
                               "</body>"
                               "</html>");
}
```

```
41 void configuracion(){  //Funcion que guarda y genera la nueva red
42   ssid=server.arg("ssid");
43   Serial.println(ssid2);
44   if (WiFi.softAP(ssid)){
45     Serial.println("Ha generado la red");
46   }else{
47     Serial.println("No se ha generado la red");
48   }
49 }
50 void gettiposnombre(){  // Funcion para obtener los nombres y los tipos de las placas conectadas
51   gettipo();
52   getnombres();
53   paginawebprincipal();
54 }
55 // Las funciones data1-8 sirven para controlar los reles
56 void data1(){
57   conectrele(0);
58   paginawebprincipal();
59 }
60 void data2(){
61   conectrele(1);
62   paginawebprincipal();
63 }
64 void data3(){
65   conectrele(2);
66   paginawebprincipal();
67 }
68 void data4(){
69   conectrele(3);
70   paginawebprincipal();
71 }
72 void data5(){
73   conectrele(4);
74   paginawebprincipal();
75 }
76 void data6(){
77   conectrele(5);
78   paginawebprincipal();
79 }
80 void data7(){
81   conectrele(6);
82   paginawebprincipal();
83 }
84 void data8(){
85   conectrele(7);
86   paginawebprincipal();
87 }
```

```
88  void cambiarnombre(){ //Funcion que genera la pagina web para cambiar los nombres de las placas
89    String c="";
90    c+="<!DOCTYPE html>";
91    c+="<html>";
92    c+="<head>";
93    c+="<title>Configuración</title>";
94    c+="<meta charset='UTF-8'>";
95    c+="</head>";
96    c+="<body>";
97    c+="</form>";
98    c+="<form action='nombres' method='get'>";
99    for(int a=0;a<WiFi.softAPgetStationNum();a++){
100     c+=nombres[a];
101     c+=":<br>";
102     c+="<input class='input1' name='nombre";
103     c+=(a-1);
104     c+="' type='text'><br>";
105   }
106   c+="<input class='boton' type='submit' value='GUARDAR'/><br><br>";
107   c+="</form>";
108   c+="</body>";
109   c+="</html>";
110   server.send(200,"text/html",c);
111 }
112 void nombre(){ // Funcion para guardar los nuevos nombres y mandarlos a las placas
113   for (int a=0;a<WiFi.softAPgetStationNum();a++){
114     String c,aux;
115     c+="nombre";
116     c+=(a-1);
117     aux=server.arg(c);
118     if (aux!=""&&aux!=" "){
119       nombres[a]=aux;
120       setnombre(a);
121     }
122   }
123   paginawebprincipal();
124 }
```

```
125 void dht(){ // Funcion para obtener la temperatura y la humedad y mostrarla en una pagina web
126   Serial.println("DHT");
127   String aux,c;
128   int aux2;
129   for(int a=0;a<WiFi.softAPgetStationNum();a++){
130     if(tipo[a]==2){
131       conectdht(a);
132       delay(100);
133       c="<!DOCTYPE html>"
134          "<html>"
135          "<head>"
136          "<title>Temperatura</title>"
137          "<meta charset='UTF-8'>"
138          "</head>"
139          "<body>"
140          "<br><br>Temperatura: ";
141        c+=temperatura;
142        c+="  Humedad: ";
143        c+=humedad;
144        c+="  Sensación termica: ";
145        c+=temperaturaAparente;
146        c+="</form>"
147           "<form action='dht' method='get'>"
148           "<input class='boton' type='submit' value='Recargar'/><br><br>"
149           "</form>"
150           "<br><form action='general' method='get'>"
151           "<input class='boton' type='submit' value='Volver'/><br><br>"
152           "</form>"
153           "</body>"
154           "</html>";
155       server.send(200,"text/html",c);
156     }
157   }
158 }
159 void telefonillo(){ // Funcion para controlar el telefonillo
160   for(int a=0;a<WiFi.softAPgetStationNum();a++){
161     if(tipo[a]==4){
162       conectrele(a);
163       paginawebprincipal();
164     }
165   }
166 }
```

```
167 void subir_persiana(){ // Funcion para subir las persianas
168   Serial.println("Subir persiana");
169   for(int a=0;a<WiFi.softAPgetStationNum();a++){
170     if(tipo[a]==3){
171       String connection=ips[a];
172       String aux;
173       aux="HOST: ";
174       aux+=connection;
175       WiFiClient client;
176       Serial.println(connection);
177       if(client.connect(connection,80)<0){
178         Serial.println("fallo en el envio");
179       }else{
180         client.println("GET /subir HTTP/1.0");
181         client.println(aux);
182         Serial.println("enviado");
183         client.stop();
184       }
185     }
186   }
187   paginawebprincipal();
188 }
189 void bajar_persiana(){ // Funcion para bajar las persianas
190   Serial.println("Bajar persiana");
191   for(int a=0;a<WiFi.softAPgetStationNum();a++){
192     if(tipo[a]==3){
193       String connection=ips[a];
194       String aux;
195       aux="HOST: ";
196       aux+=connection;
197       WiFiClient client;
198       Serial.println(connection);
199       if(client.connect(connection,80)<0){
200         Serial.println("fallo en el envio");
201       }else{
202         client.println("GET /bajar HTTP/1.0");
203         client.println(aux);
204         Serial.println("enviado");
205         client.stop();
206       }
207     }
208   }
209   paginawebprincipal();
210 }
```

```
211 void parar_persiana(){ // Funcion para parar las persianas
212   Serial.println("Parar persiana");
213   for(int a=0;a<WiFi.softAPgetStationNum();a++){
214     if(tipo[a]==3){
215       String connection=ips[a];
216       String aux;
217       aux="HOST: ";
218       aux+=connection;
219       WiFiClient client;
220       Serial.println(connection);
221       if(client.connect(connection,80)<0){
222         Serial.println("fallo en el envio");
223       }else{
224         client.println("GET /parar HTTP/1.0");
225         client.println(aux);
226         Serial.println("enviado");
227         client.stop();
228       }
229     }
230   }
231   paginawebprincipal();
232 }
233 void encender(){ // Funcion para encender el termostato
234   Serial.println("Encender termostato");
235   for(int a=0;a<WiFi.softAPgetStationNum();a++){
236     if(tipo[a]==5){
237       String connection=ips[a];
238       String aux;
239       aux="HOST: ";
240       aux+=connection;
241       WiFiClient client;
242       Serial.println(connection);
243       if(client.connect(connection,80)<0){
244         Serial.println("fallo en el envio");
245       }else{
246         client.println("GET /encender HTTP/1.0");
247         client.println(aux);
248         Serial.println("enviado");
249         client.stop();
250       }
251     }
252   }
253   paginawebprincipal();
254 }
```

```
255 void apagar(){ // Funcion para apagar el termostato
256   Serial.println("Encender termostato");
257   for(int a=0;a<WiFi.softAPgetStationNum();a++){
258     if(tipo[a]==5){
259       String connection=ips[a];
260       String aux;
261       aux="HOST: ";
262       aux+=connection;
263       WiFiClient client;
264       Serial.println(connection);
265       if(client.connect(connection,80)<0){
266         Serial.println("fallo en el envio");
267       }else{
268         client.println("GET /apagar HTTP/1.0");
269         client.println(aux);
270         Serial.println("enviado");
271         client.stop();
272       }
273     }
274   }
275   paginawebprincipal();
276 }
```

```
277 void setup() {
278   Serial.begin(115200);
279   WiFi.mode(WIFI_AP);
280   if (WiFi.softAP(ssid)){
281     Serial.println("Ha generado la red");
282   }else{
283     Serial.println("No se ha generado la red");
284   }
285   server.on("/",conecxion);
286   server.on("/general",general);
287   server.on("/conf",conf);
288   server.on("/guardar_conf",configuracion);
289   server.on("/gettiponombre",gettiposnombre);
290   server.on(funcion[0],data1);
291   server.on(funcion[1],data2);
292   server.on(funcion[2],data3);
293   server.on(funcion[3],data4);
294   server.on(funcion[4],data5);
295   server.on(funcion[5],data6);
296   server.on(funcion[6],data7);
297   server.on(funcion[7],data8);
298   server.on("/cambiar_nombre",cambiarnombre);
299   server.on("/nombres",nombre);
300   server.on("/dht",dht);
301   server.on("/telefonillo",telefonillo);
302   server.on("/subir",subir_persiana);
303   server.on("/bajar",bajar_persiana);
304   server.on("/parar",parar_persiana);
305   server.on("/encender",encender);
306   server.on("/apagar",apagar);
307   server.begin();
308 }
309 void loop() {
310   server.handleClient();
311 }
```

```
312 void paginawebprincipal(){ // Funcion que genera la pagina web principal en funcion de las placas conectadas
313   String c;
314   c+="<!DOCTYPE html>";
315   c+="<html>";
316   c+="<head>";
317   c+="<title>Control Domótica</title>";
318   c+="<meta charset='UTF-8'>";
319   c+="</head>";
320   c+="<body>";
321   c+="</form><br>";
322   for(int a=0;a<WiFi.softAPgetStationNum();a++){
323     switch(tipo[a]){
324       case 1:{
325         c+="<br><form action='";
326         c+=boton[a];
327         c+="' method='get'>";
328         c+="<input class='boton' type='submit' value='";
329         c+=nombres[a];
330         c+="'/><br>";
331         c+="</form>";
332         break;
333       }
334       case 2:{
335         c+="<br><form action='";
336         c+="dht";
337         c+="' method='get'>";
338         c+="<input class='boton' type='submit' value='";
339         c+=nombres[a];
340         c+="'/><br>";
341         c+="</form>";
342         break;
343       }
```

```
344       case 3:{
345         c+=nombres[a];
346         c+="<br><form action='";
347         c+="subir";
348         c+="' method='get'>";
349         c+="<input class='boton' type='submit' value='";
350         c+="Subir persiana";
351         c+="'/><br><br>";
352         c+="</form>";
353         c+="<form action='";
354         c+="bajar";
355         c+="' method='get'>";
356         c+="<input class='boton' type='submit' value='";
357         c+="Bajar persiana";
358         c+="'/><br><br>";
359         c+="</form>";
360         c+="<form action='";
361         c+="parar";
362         c+="' method='get'>";
363         c+="<input class='boton' type='submit' value='";
364         c+="Parar persiana";
365         c+="'/><br>";
366         c+="</form>";
367         break;
368       }
369       case 4:{
370         c+="<br><form action='";
371         c+="telefonillo";
372         c+="' method='get'>";
373         c+="<input class='boton' type='submit' value='";
374         c+=nombres[a];
375         c+="'/><br>";
376         c+="</form>";
377         break;
378       }
```

```
      case 5:{
        c+=nombres[a];
        c+="<br><form action='";
        c+="encender";
        c+="' method='get'>";
        c+="<input class='boton' type='submit' value='";
        c+="Encender termostato";
        c+="'/><br><br>";
        c+="</form>";
        c+="<form action='";
        c+="apagar";
        c+="' method='get'>";
        c+="<input class='boton' type='submit' value='";
        c+="Apagar termostato";
        c+="'/><br>";
        c+="</form>";
        break;
      }
      case 6:{
        c+="<a href=http://";
        c+=ips[a];
        c+="/general>";
        c+=nombres[a];
        c+="</a>";
        break;
      }
    }
  }
  c+="<br><br><form action='cambiar_nombre' method='get'>";
  c+="<input class='boton' type='submit' value='Cambiar nombres'/><br>";
  c+="</form>";
  c+="<br><form action='conf' method='get'>";
  c+="<input class='boton' type='submit' value='Cambiar red domestica'/><br>";
  c+="</form>";
  c+="<br><form action='gettiponombre' method='get'>";
  c+="<input class='boton' type='submit' value='Get tipos y nombres'/><br>";
  c+="</form>";
  c+="</body>";
  c+="</html>";
  server.send(200,"text/html",c);
  Serial.println("pagina enviada");
}
```

```
421 void gettipo(){ // Funcion para pedir a las placas su tipo
422   Serial.println("Gettipo");
423    for(int a=0;a<WiFi.softAPgetStationNum();a++){
424       Serial.println("conect");
425       String connection=ips[a];
426       String aux,line="";
427       aux="HOST: ";
428       aux+=connection;
429       WiFiClient client;
430       Serial.println(connection);
431       if(client.connect(connection,80)<0){
432         Serial.println("fallo en el envio");
433       }else{
434         client.println("GET /tipo HTTP/1.0");
435         client.println(aux);
436         Serial.println("enviado");
437         while (client.connected()){
438           if (client.available()){
439             line += client.readStringUntil('\n');
440             Serial.print("1 ");
441             Serial.println(line);
442           }
443         }
444          int pos=line.indexOf("Datos=");
445          String line2=line.substring(pos+6);
446          Serial.print("2");
447          Serial.println(line2);
448          pos=line2.indexOf("Tipo=");
449          tipo[a]=line2.substring(pos+5).toInt();
450         client.stop();
451       }
452    }
453 }
```

```
454 void conectrele(int a){ // Funcion que conecta con el rele en la posicion que se te ha pasado
455   Serial.println("Conect rele");
456   String connection=ips[a];
457   String aux;
458   aux="HOST: ";
459   aux+=connection;
460   WiFiClient client;
461   Serial.println(connection);
462   if(client.connect(connection,80)<0){
463     Serial.println("fallo en el envio");
464   }else{
465     client.println("GET /gpio HTTP/1.0");
466     client.println(aux);
467     Serial.println("enviado");
468     client.stop();
469   }
470 }
```

```
471 void conectdht(int a){ // Funcion que conecta con el modulo DHT y le pide la temperatura y la humedad
472   Serial.println("Conect DHT");
473   String connection=ips[a];
474   String aux,line="";
475   aux="HOST: ";
476   aux+=connection;
477   WiFiClient client;
478   Serial.println(connection);
479   if(client.connect(connection,80)<0){
480     Serial.println("fallo en el envio");
481   }else{
482     client.println("GET /dht HTTP/1.0");
483     client.println(aux);
484     Serial.println("enviado");
485     while (client.connected()){
486       if (client.available()){
487         line += client.readStringUntil('\n');
488         Serial.print("1 ");
489         Serial.println(line);
490       }
491     }
492    int pos=line.indexOf("Datos=");
493    String line2=line.substring(pos+6);
494    Serial.print("2");
495    Serial.println(line2);
496    pos=line2.indexOf("Temp=");
497    temperatura=line2.substring(pos+5,pos+10).toFloat();
498    pos=line2.indexOf("Hum=");
499    humedad=line2.substring(pos+4,pos+9).toFloat();
500    pos=line2.indexOf("TempAp="),
501    temperaturaAparente=line2.substring(pos+7,pos+12).toFloat();
502    client.stop();
503   }
504 }
```

```
505 void setnombre(int a){ // Funcion que manda el nombre nuevo a la placa
506   Serial.println("Set Nombre");
507   String connection=ips[a];
508   String aux,aux2="";
509   aux="HOST: ";
510   aux+=connection;
511   WiFiClient client;
512   Serial.println(connection);
513   if(client.connect(connection,80)<0){
514     Serial.println("fallo en el envio");
515   }else{
516     aux2+="GET /setnombre?nombre=";
517     aux2+=nombres[a];
518     aux2+=" HTTP/1.0";
519     client.println(aux2);
520     client.println(aux);
521     Serial.println("enviado");
522     client.stop();
523   }
524 }
```

```
525 void getnombres(){ // Funcion que pide los nombres a todas las placas
526  Serial.println("Get Nombres");
527  for(int a=0;a<WiFi.softAPgetStationNum();a++){
528     Serial.println("conect");
529     String connection=ips[a];
530     String aux,line="";
531     aux="HOST: ";
532     aux+=connection;
533     WiFiClient client;
534     Serial.println(connection);
535     if(client.connect(connection,80)<0){
536       Serial.println("fallo en el envio");
537     }else{
538       client.println("GET /getnombre HTTP/1.0");
539       client.println(aux);
540       Serial.println("enviado");
541       while (client.connected()){
542         if (client.available()){
543           line += client.readStringUntil('\n');
544           Serial.print("1 ");
545           Serial.println(line);
546         }
547       }
548        int pos=line.indexOf("Datos=");
549        String line2=line.substring(pos+6);
550        Serial.print("2");
551        Serial.println(line2);
552        pos=line2.indexOf("Nombre=");
553        Serial.print("Nombre: ");
554        Serial.println(line2.substring(pos+7));
555        nombres[a]=line2.substring(pos+7);
556       client.stop();
557     }
558  }
559 }
```